Université de Montréal

Etude de la traduction automatique des bulletins météorologiques

par

Thomas Leplus

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté à la Faculté des études supérieures
en vue de l'obtention du grade de Maîtrise
en Informatique

Novembre 2004

Université de Montréal
Faculté des études supérieures

Ce mémoire intitulé :

Etude de la traduction automatique des bulletins météorologiques

présenté par :

Thomas Leplus

a été évalué par un jury composé des personnes suivantes :

Balázs Kégl	président-rapporteur
Philippe Langlais	directeur de recherche
Guy Lapalme	codirecteur
Petko Valtchev	membre du jury

Abstract

For more than 30 years, researchers have been interested in the translation of weather forecasts. The simplicity of the language seen in those forecasts and the large quantity of text that has to be translated daily make it an ideal application for the field of machine translation. The first system designed in the 70's at the Université de Montréal is still one of the most famous successes of machine translation. However one can wonder if the approaches available today could not allow us to improve the machine translation of weather forecasts. We have studied the potential of several approaches like translation memories and statistical machine translation. We show the pros and cons of each approach and we put them to application by creating a new system of machine translation for weather forecasts.

Keywords: machine translation, translation memory, statistical translation, meteorology, weather forecast, weather report.

Table des matières

Liste des tableaux

Table des figures

Abréviations

Au cours de ce mémoire, nous utiliserons parfois les sigles, abréviations et symboles suivants :

ADN : Acide Désoxyribonucléique
AFD : Automate Fini Déterministe
AFDP : Automate Fini Déterministe Pondéré
BLEU : *Bilingual Evaluation Understudy*
CMC : Centre Météorologique Canadien
EC : Environnement Canada
EM : *Expectation Maximization*
Go : giga-octets
IBM : *International Business Machines Corporation*
Ko : kilo-octets
Mo : méga-octets
MT : *Machine Translation*
NIST : *National Institute of Standards and Technology*
PBM : *Phrase-Based Model*
RALI : Recherche Appliquée en Linguistique Informatique
RAM : *Random-Access Memory*
ROM : *Read-Only Memory*
SER : *Sentence Error Rate*
SMT : *Statistical Machine Translation*
TAO : Traduction Assistée par Ordinateur
TAUM : Traduction Automatique à l'Université de Montréal
WBM : *Word-Based Model*
WER : *Word Error Rate*

Remerciements

Je tiens à remercier mon directeur de recherche, Philippe Langlais, et mon codirecteur, Guy Lapalme, pour leur collaboration, leurs conseils et leurs critiques constructives tout au long de mon travail. Je remercie aussi Elliott Macklovitch pour toutes les informations qu'il m'a apporté. Je suis très reconnaissant envers tous les membres de l'équipe du RALI pour le séjour inoubliable que j'ai passé parmi eux. Efin je remercie Rick Jones du Centre Météorologique Canadien d'avoir permis au RALI d'obtenir les données nécessaires à cette étude.

Introduction

Entre 1975 et 1976, le groupe de Traduction Automatique de l'Université de Montréal (TAUM) a développé un système de traduction automatique commandité par le gouvernement du Canada et dont la fonction est de traduire des bulletins météorologiques de l'anglais vers le français. Ce système, connu sous le nom de TAUM-METEO, était entièrement basé sur des règles systématiques : des lexiques bilingues, des grammaires et d'autres règles pour des fonctions linguistiques spécifiques comme le réordonnancement des mots ou l'élision des articles. Le tout était implanté dans le métalangage système-Q. Le système ne fut jamais mis en exploitation par le groupe TAUM qui s'était entre temps désintéressé du projet pour un autre : TAUM-AVIATION.

C'est en 1978 que John CHANDIOUX, un ancien membre du groupe TAUM, reprend le projet METEO pour le compte du gouvernement canadien. Son nouveau système, METEO-2, atteint un "taux de succès de 80%"[1] [8]. Le système pouvait traduire près de 7 000 mots par jour sur une station Cyber-7600 équipée de 1,6 Mo de RAM.

[1] 80% des phrases produites par le système ont été jugées correctes par un traducteur humain.

A partir de 1984, le système METEO-2 est exploité en continu par le Centre Météorologique Canadien (CMC) d'Environnement Canada. METEO-2 est une refonte du système METEO basée sur GramR[2], un langage de programmation linguistique développé par CHANDIOUX. En 1990, le système traduisait environ 45 000 mots par jour. C'était un des rares systèmes de traduction au monde à produire des sorties qui étaient rendues accessibles au public[3] sans nécessiter de corrections humaines (même si,

pour des raisons de responsabilité légale, des traducteurs humains validaient les traductions produites).

[2] GramR et meteo sont des marques déposées et sont la propriété de John Chandioux.

[3] http://meteo.ec.gc.ca/forecast/textforecast_f.html

Une des raisons possibles du succès qu'a rencontré ce système est la répétitivité de la tâche de traduction des bulletins météorologiques (travail particulièrement pénible pour des traducteurs humains). Un autre facteur important est la courte durée de vie de ces bulletins qui demande une traduction presque en temps réel, de jour comme de nuit. Il aurait fallu beaucoup de traducteurs humains pour assurer une telle disponibilité.

Les systèmes commerciaux actuels de traduction automatique (*Systran*, *Power Translator*...) se basent toujours sur un moteur de traduction qui utilise des lexiques et des grammaires de plus en plus complexes. L'avantage de cette approche est que le système peut être mis-à-jour régulièrement par de nouvelles entrées dans les lexiques et de nouvelles règles dans les grammaires.

Une autre approche intuitive de la traduction automatique est l'utilisation d'une mémoire de traduction. L'idée est de réutiliser des traductions antérieures pour ne pas refaire plusieurs fois le même travail de traduction. Les premiers systèmes utilisaient des mémoires contenant des phrases entières et leurs traductions. Nous étudierons la traduction par une mémoire phrastique au chapitre 2. Nous montrerons que cette approche est très efficace pour la traduction de bulletins météorologiques mais pour la plupart des autres textes, nous savons que ce n'est pas le cas car le taux de répétition des phrases dans les textes est souvent très bas. Pour cette raison, différentes techniques ont été proposées comme celle d'utiliser une mémoire sous-phrastique, c'est-à-dire une mémoire de traduction des groupes de mots et non

plus des phrases entières. D’autres techniques encore ont été suggérées. Au chapitre 3, nous étudierons par exemple une technique de consensus entre les différentes traductions approximatives produites par une mémoire de traduction phrastique.

Depuis la création du système METEO, l’état de l’art en traduction automatique a toutefois considérablement évolué. Le changement est venu dans les années 80 quand des laboratoires comme ceux d’IBM ont commencé à s’orienter vers une approche empirique de la traduction automatique. Comme de plus en plus de données linguistiques bilingues sont devenues disponibles à la communauté scientifique, les chercheurs se sont intéressé aux techniques basées sur l’exemple et l’apprentissage. Les travaux de l’époque sur la reconnaissance de la parole ont fait le rapprochement entre le traitement de la langue et celui de l’information, offrant ainsi un nouveau cadre à la traduction automatique : la théorie de la communication, une théorie mathématique fondée sur les statistiques. Nous reviendrons plus en détails sur les principes de cette théorie et ses applications en traduction automatique au chapitre 4.

L’évolution du domaine du traitement de la langue n’est pas le seul changement dont va profiter notre étude. Les techniques de programmation ont également été révolutionnées, que ce soit par la création de nouveaux langages particulièrement propices au traitement du texte comme Perl ou plus généralement par l’avènement de nouveaux paradigmes comme la programmation orientée objet. Toutes ces nouvelles techniques vont considérablement faciliter notre travail de développement.

Les ressources disponibles aujourd’hui pour les linguistes informatiques sont aussi bien plus nombreuses : qu’il s’agisse de ressources logicielles (comme la multitude

d'outils de traitement de la langue disponibles sur Internet) ou matérielles (comme les nouvelles générations de microprocesseurs 64 bits et les espaces de mémoire RAM et ROM de plusieurs giga-octets). Nous comptons profiter de toutes ces ressources tout au long de notre étude.

Enfin, nous disposons d'une nouvelle ressource qui nous sera essentielle pour notre projet : nous avons reçu du CMC deux années de traductions de bulletins météorologiques. Nous allons donc utiliser ces données pour construire un corpus bilingue de bulletins météorologiques afin de pouvoir comparer les résultats des techniques de traduction automatique les plus récentes. A partir des résultats de notre étude, nous espérons concevoir un système de traduction complet et performant.

Chapitre 1
Préparation du bitexte

Dans les chapitres suivants, nous allons étudier différentes approches basées sur l'apprentissage. Nous allons essayer de capturer automatiquement des connaissances en vue de créer un système de traduction qui exploitera ces connaissances pour produire ses propres traductions. Il nous faut donc une source de connaissances dans un format approprié à nos besoins. Dans le cas de la traduction automatique, cette source de connaissance est souvent un corpus bilingue aligné (aussi appelé bitexte), c'est-à-dire un ensemble de textes dont les segments en relation de traduction sont identifiés. Il existe différents niveaux d'alignement. Les textes peuvent être alignés par exemple au niveau des paragraphes, des phrases ou bien des mots. Pour notre étude, nous aurons besoin d'un grand nombre de paires de phrases anglaises et françaises en relation de traduction. Nous allons donc construire un bitexte aligné au niveau des phrases. Si nécessaire, nous pourrons toujours par la suite aligner les mots au sein de chaque paire de phrases pour obtenir un alignement au niveau des mots.

Dans ce chapitre, nous étudions les données qui nous ont été fournies par le CMC et nous justifions comment nous avons extrait des paires de phrases en relation de traduction. Nous montrons que cette tâche qui nous semblait simple *a priori* nécessite en fait une analyse soigneuse des données et une préparation attentive de leurs traitements car toute erreur dans le bitexte aura un impact direct sur les résultats.

1.1 Analyse du format des données

Les données météorologiques qui nous ont été fournies par le CMC sont les archives des bulletins émis en 2002 et 2003. Ces bulletins sont regroupés dans des fichiers correspondant chacun à une période de 6 heures (soit 4 fichiers par jour). Le tout représente un total de 561 Mo de texte. Nous ne disposions d'aucune autre information sur ces données ainsi notre premier travail a-t'il consisté à étudier les fichiers pour comprendre comment les données y sont présentées. La figure 1.1 montre un extrait du fichier pour le 1er janvier 2002. Un même fichier contient à la fois des bulletins en anglais et en français. Les bulletins en relation de traduction sont généralement à proximité dans les fichiers mais ce n'est pas systématique.

FPCN78 CWXK 312130
RESUME DES PREVISIONS POUR L'EST DU QUEBEC EMISES
PAR ENVIRONNEMENT CANADA RIMOUSKI A 16H30 HNE LE
LUNDI 31 DECEMBRE 2001 POUR MARDI LE 01 JANVIER
2002. PLUTOT NUAGEUX AVEC AVERSES DE NEIGE. MAX
PRES DE MOINS 9. VENTEUX. PROBABILITE DE
PRECIPITATIONS 70 POUR CENT.
FIN

FPCN78 CWUL 312130
RESUME DES PREVISIONS POUR L'OUEST DU QUEBEC EMISES
PAR ENVIRONNEMENT CANADA MONTREAL A 16H30 HNE LE LUNDI
31 DECEMBRE 2001 POUR MARDI LE 01 JANVIER 2002. CIEL
VARIABLE AVEC AVERSES DE NEIGE. MAX PRES DE MOINS 7.
FIN

FPCN18 CWUL 312130
SUMMARY FORECAST FOR WESTERN QUEBEC ISSUED BY
ENVIRONMENT CANADA MONTREAL AT 4.30 PM EST MONDAY 31
DECEMBER 2001 FOR TUESDAY 01 JANUARY 2002. VARIABLE
CLOUDINESS WITH FLURRIES. HIGH NEAR MINUS 7.
END

FPCN18 CWXK 312130
SUMMARY FORECAST FOR EASTERN QUEBEC ISSUED BY
ENVIRONMENT CANADA RIMOUSKI AT 4.30 PM EST MONDAY 31
DECEMBER 2001 FOR TUESDAY 01 JANUARY 2002. MOSTLY
CLOUDY WITH FLURRIES. HIGH NEAR MINUS 9. WINDY.
PROBABILITY OF PRECIPITATION 70 PERCENT.
END

FPCN78 CWQB 312130
RESUME DES PREVISIONS POUR LE CENTRE DU QUEBEC EMISES
PAR ENVIRONNEMENT CANADA QUEBEC A 16H30 HNE LE LUNDI
31 DECEMBRE 2001 POUR MARDI LE 01 JANVIER 2002. CIEL
VARIABLE. MAX PRES DE MOINS 9.
FIN

FPCN18 CWQB 312130
SUMMARY FORECAST FOR CENTRAL QUEBEC ISSUED BY
ENVIRONMENT CANADA QUEBEC AT 4.30 PM EST MONDAY 31
DECEMBER 2001 FOR TUESDAY 01 JANUARY 2002. VARIABLE

```
CLOUDINESS. HIGH NEAR MINUS 9.
END
```

Figure 1.1: Extrait du fichier du 1er janvier 2002.

Nous remarquons immédiatement que les bulletins sont entièrement en majuscules non accentuées. Cette pratique typographique est nécessaire parce que les bulletins sont transmis dans le monde entier et le jeu de caractères recommandé par le standard des communications électroniques internationales se limite aux majuscules non accentuées, l'espace et quelques caractères de ponctuation. Le but de cette contrainte était de faciliter la compression des textes à une époque où la bande passante des réseaux de communication était limitée. Aujourd'hui, ce standard persiste pour maintenir la compatibilité avec d'anciens systèmes toujours en exploitation dans certaines parties du monde.

Chaque bulletin commence par une ligne unique d'identification telle que `FPCN78 CWUL 312130`. Nous avons trouvé sur le site Internet du CMC[1] que le code `FPCN78 CWUL` correspond aux bulletins émis en français par le bureau de Montréal et qui couvrent la région de Montréal et l'ouest du Québec. Le code `FPCN18 CWUL` correspond aux mêmes bulletins mais traduits en anglais. Le chiffre suivant le code est un horodatage : `312130` signifie que le bulletin à été émis le 31 décembre à 21h30. Il se trouve dans le fichier du 1er janvier en raison du délai de traduction.

[1] `http://www.smc-msc.ec.gc.ca/cmc/index_f.html`

Nb.	Code	Sujet
4446	FPCN24 CWHX	Prévisions Techniques - Maritimes (anglais)
4442	FPCN23 CWHX	Prévisions Navtex - Ouest des Maritimes (anglais)
4438	FPCN83 CWHX	Prévisions Navtex - Ouest des Maritimes (français)
4430	FPCN84 CWHX	Prévisions Techniques - Maritimes (français)
3703	FPCN24 CYQX	Prévisions Navtex - Est de Terre-Neuve (anglais)
3693	FPCN84 CYQX	Prévisions Navtex - Est de Terre-Neuve (français)
3684	FPCN26 CYQX	Prévisions Marines - Port d'Halifax (anglais)
3681	FPCN25 CYQX	Prévisions Navtex - Labrador (anglais)
3676	FPCN86 CYQX	Prévisions Marines - Port d'Halifax (français)
3670	FPCN85 CYQX	Prévisions Navtex - Labrador (français)

Table 1.1: Les codes de bulletins les plus fréquents.

La table 1.1 indique les codes les plus fréquents avec leur fréquence, leur sujet et leur langue. Nous observons que les codes vont par paires : les bulletins ayant les codes `FPCN24 CWHX` et `FPCN84 CWHX` par exemple

portent sur le même sujet, seule la langue change. Le nombre d'occurrences de ces deux codes est presque identique, ce qui semble indiquer que les bulletins étiquetés FPCN24 CWHX sont les traductions en anglais des bulletins FPCN84 CWHX. La lecture de plusieurs de ces bulletins nous confirme cette hypothèse. Il en va de même pour les codes FPCN25 CWHX et FPCN85 CWHX ou encore FPCN24 CYQX et FPCN84 CYQX... En examinant d'avantage les bulletins, nous concluons que tous les bulletins qui débutent par un code entre FPCN10 et FPCN29 sont en relation de traduction avec les bulletins ayant la même en-tête mais avec respectivement les codes entre FPCN70 et FPCN89. Les premiers sont les bulletins en anglais et les seconds ceux en français.

Les bulletins en français se terminent tous par une ligne commençant par le mot FIN alors que ceux en anglais se terminent par une ligne commençant par END. Nous avons donc un deuxième critère simple pour déterminer la langue d'un bulletin. Le reste de la dernière ligne de chaque bulletin semble être une signature identifiant l'auteur du bulletin. Le plus souvent, il s'agit d'un nom propre ou d'initiales. Le tableau 1.2 montre les fins de bulletins les plus fréquentes.

Nb.	Fin
32426	END
27958	FIN
11036	FIN/
7635	END/ARWC
5522	END/
5403	FIN/ARWC

Table 1.2: Les fins de bulletins les plus fréquentes.

1.2 Traitement des données

Pour construire notre bitexte aligné au niveau des phrases, nous devons déterminer le début et la fin de chaque bulletin, puis trier ces bulletins selon leur langue et les apparier. Enfin nous devons identifier les phrases en relation de traduction dans chaque paire de bulletins.

La première étape des traitements consiste à séparer les bulletins en fichiers individuels. Grâce à l'étude que nous avons faite précédemment sur le format des données, un simple script Perl va nous suffire pour réaliser cette tâche. Notre script identifie le début et la fin de chaque bulletin et le place dans un fichier nommé selon son code. Par la même occasion, ce script utilise le marqueur de fin de bulletin pour confirmer la langue des bulletins et les répartir dans deux arborescences de dossiers

différentes : une pour l’anglais et une pour le français. A la fin de cette étape, il nous reste 125 025 bulletins en français et 148 822 bulletins en anglais (soit 12% de moins que les 309 677 bulletins d’origine).

La seconde étape consiste à apparier les bulletins en relation de traduction. Nous avons montré que les bulletins qui débutent par un code entre `FPCN10` et `FPCN29` sont les traductions de ceux ayant la même en-tête mais avec respectivement les codes entre `FPCN70` et `FPCN89`. Un autre script Perl rassemble dans une même arborescence ces paires de bulletins. Nous obtenons ainsi 89 697 paires de bulletins, ce qui est tout à fait satisfaisant vu la simplicité de la règle utilisée pour les apparier. Il nous reste 35 328 bulletins en français et 59 125 bulletins en anglais non appariés. Il y a donc au moins 23 797 bulletins en anglais dont nous n’avons pas la traduction. Probablement d’avantage si l’on suppose qu’il nous manque aussi la traduction de certains bulletins en français.

Il est possible de réduire le nombre de bulletins perdus à cette étape en cherchant d’autres règles pour les codes de bulletins restants. Nous pourrions également utiliser un logiciel d’alignement de textes. Un tel logiciel est capable de reconnaître des textes qui sont probablement en relation de traduction grâce à un lexique bilingue et en observant les occurrences de certains invariants de traduction comme les chiffres. Nous avons jugé que cela ne valait pas la peine de risquer d’introduire des erreurs d’appariement dans notre bitexte alors que nous avons déjà suffisamment de paires de bulletins.

Ensuite nous devons découper chaque bulletin en phrases et chaque phrase en mots. Cette opération de tokenisation peut sembler triviale au premier abord mais il ne suffit pas de considérer qu’un point termine chaque phrase et qu’une espace[2] sépare chaque mot. Il y a des situations plus complexes comme par exemple le cas de l’apostrophe : `D’OUEST` est composé de deux mots distincts alors que `AUJOURD’HUI` n’est qu’un seul mot. De plus, le même caractère apostrophe est parfois utilisé comme guillemet.

[2] Le nom espace est féminin quand il désigne le caractère typographique.

La tokenisation est simple conceptuellement mais elle s’avère un problème délicat en pratique pour lequel de nombreux chercheurs continuent de proposer des solutions [7]. Un programme de tokenisation dispose en général de nombreuses règles de segmentation des phrases et aussi de listes d’exceptions. Nous avons choisi d’utiliser le tokeniseur crée au laboratoire RALI par George Foster. Après inspection des fichiers produits par le logiciel, nous avons constaté que ce tokeniseur produit des résultats satisfaisants sans aucune configuration particulière à notre bitexte. Pourtant, le fait par exemple que notre corpus soit entièrement en

majuscules non accentuées aurait pu empêcher le tokeniseur de reconnaître les exceptions ou encore lui faire interpréter les caractères `..` comme une fin de phrase alors qu'ils sont utilisés dans les bulletins à la place du caractère : (comme par exemple dans la phrase `CE MATIN .. AVERSES DE PLUIE`).

En fait, la seule erreur du tokeniseur que nous avons remarquée est liée au fait que les textes des bulletins n'utilisent pas de point après les abréviations (comme `MM`, `CM` ou `KM`). Lorsqu'un point se trouve après une abréviation (par exemple `MM.`), il s'agit bel et bien d'une fin de phrase contrairement à ce que notre tokeniseur suggère. Pour corriger ce problème, nous avons créé un script qui passe à travers tous les fichiers tokenisés pour ajouter un token de fin de phrase (`{sent}`) après chaque point qui ne précède pas déjà un tel token. Pour chaque bulletin, nous avons donc maintenant un fichier contenant un token par ligne (un token étant soit un mot, soit un élément de ponctuation, soit un marqueur de fin de phrase).

Enfin nous avons réalisé plusieurs post-traitements comme par exemple celui de vérifier que toutes les lettres sont bien en majuscules car nous avons remarqué par inspection quelques exceptions. Nous avons aussi converti les heures dans les fichiers anglais au format 24 heures. Par la même occasion, nous nous sommes assuré que, dans tous les fichiers, les jours, les heures et les minutes sont toujours indiqués avec deux décimales. Tous ces traitements ont encore une fois été réalisés à l'aide de scripts Perl.

Une fois les bulletins tokenisés, il nous reste à aligner les phrases des bulletins. Ce travail a été réalisé par l'aligneur de phrases JAPA créé par Philippe LANGLAIS [15]. Cet aligneur identifie les phrases en relation de traduction au sein d'un bulletin en exploitant un lexique bilingue ainsi que d'autres indices tels que la longueur des phrases ou les occurrences des invariants de traduction (comme les expressions numériques et certaines entités nommées). Parfois une phrase anglaise peut être traduite par plusieurs phrases anglaises et inversement. L'aligneur ne nous donne donc pas que des relations "une à une" entre des phrases. Pour simplifier l'usage de notre bitexte, nous n'avons toutefois conservé que les phrases anglaises qui sont alignées à une seule phrase française. Nous avons ainsi obtenu un bitexte de 4 346 684 paires de phrases alignées. Il s'agit d'un gros corpus comparé à d'autre corpus usuels comme le corpus HANSARD des débats parlementaires canadiens qui comporte environ 1,7 million de paires de phrases.

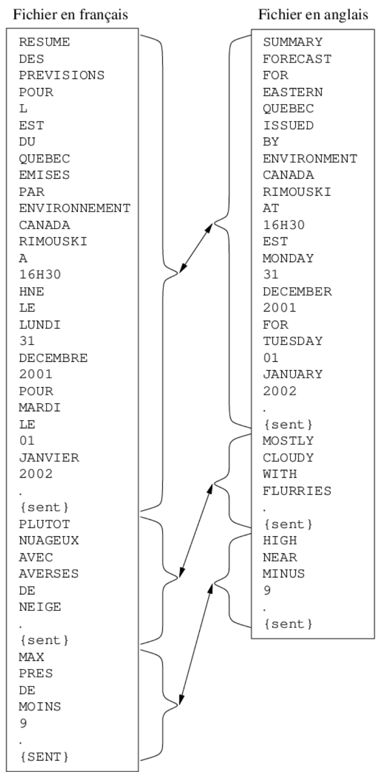


FIGURE 1.2: Extrait du bitexte avec les phrases tokenisées et alignées.

La figure 1.2 présente un extrait de notre bitexte. La colonne de gauche est extraite du fichier en français pour le 1er janvier 2002 et la colonne de droite est la partie correspondante du fichier en anglais. Nous avons également besoin d'un fichier qui donne les relations de traduction entre les phrases (représentés par des flèches sur la figure 1.2) mais comme nous n'avons gardé que les paires de phrases “une à une”, ce fichier est très simple : la première phrase du fichier anglais est alignée avec la première du fichier français, la seconde du fichier anglais avec la seconde du fichier français et ainsi de suite.

Au final, cette étape préparatoire commune à toutes les approches basées sur des bitextes a nécessité l'écriture de près de 4 000 lignes de script et plusieurs itérations contrôlées de tout le processus ont été nécessaires avant de parvenir à obtenir un bitexte sans erreur apparente.

Maintenant que nous avons isolé, tokenisé et apparié nos bulletins, nous pouvons en étudier les propriétés linguistiques.

1.3 Analyse linguistique

Les bulletins sont des textes relativement courts : ils comportent en moyenne environ 300 mots. En lisant l'extrait de bitexte de la figure 1.2, nous voyons que le style des bulletins est télégraphique : il n'y a presque aucune ponctuation et les articles et pronoms sont très rares. Le format répétitif des bulletins pourrait laisser penser que le logiciel utilisé par les météorologues pour saisir les bulletins impose un certain format ou une certaine syntaxe. En fait, le CMC recommande seulement à ses météorologues de respecter une certaine nomenclature (comme d'utiliser toujours certaines expressions plutôt que d'autres comme par exemple `VENTS LEGERS` et non `VENTS FAIBLES`). Les météorologues restent libres de saisir les bulletins comme ils le veulent (avec toutes les variations de style, de grammaire et d'orthographe que cela implique). Il ne faut donc pas se laisser tromper par l'apparente monotonicité du texte et nous montrerons au chapitre 2 qu'un système qui se contente de mémoriser la traduction de chaque phrase rencontrée dans le bitexte a des performances insuffisantes.

Pour ne pas biaiser nos futures expériences, nous avons suivi la pratique courante dans le domaine de la traduction statistique consistant à diviser notre bitexte en trois sous-bitextes :

- TRAIN comporte les bulletins datés de janvier 2001 à octobre 2002. Il sera notre bitexte d'entraînement.
- BLANC comporte les bulletins du mois de décembre 2002. Il sera utilisé pour régler les paramètres des différentes approches que nous allons essayer.
- TEST comporte les bulletins du mois de novembre 2002. Il servira à évaluer notre système final.

Nous avons découpé notre corpus de manière chronologique afin de simuler autant que possible les conditions réelles d'utilisation : notre système sera utilisé pour traduire des bulletins postérieurs à ceux utilisés pour son entraînement.

	Anglais			Français		
bitexte	phrases	mots	tokens	phrases	mots	tokens
TRAIN	4 188 100	30 325 726	9 980	4 188 100	37 329 793	11 104
BLANC	122 356	887 547	3 011	122 356	1 092 345	3 246
TEST	36 228	268 822	1 864	36 228	333 394	1 983
total	4 346 684	31 482 095	10 126	4 346 684	38 755 541	11 256

TABLE 1.3: Principales caractéristiques de notre bitexte.

La table 1.3 présente les principales caractéristiques de nos bitextes : le nombre de phrases, de mots et de tokens (mots différents) dans chaque langue. Notons que le vocabulaire français (11 256 tokens) est environ 11% plus grand que le vocabulaire anglais (10 126 tokens) comme c'est souvent le cas dans les bitextes français-anglais. De même, les phrases françaises sont en moyenne plus longues (8,9 mots) que les phrases anglaises (7,2 mots). Par contre, les phrases sont en moyenne deux fois plus courtes que dans un corpus comme le HANSARD, ce qui s'explique par le style télégraphique des bulletins météorologiques. Les distributions des longueurs de phrases et de mots selon la langue sont représentées aux figures 1.3 et 1.4.

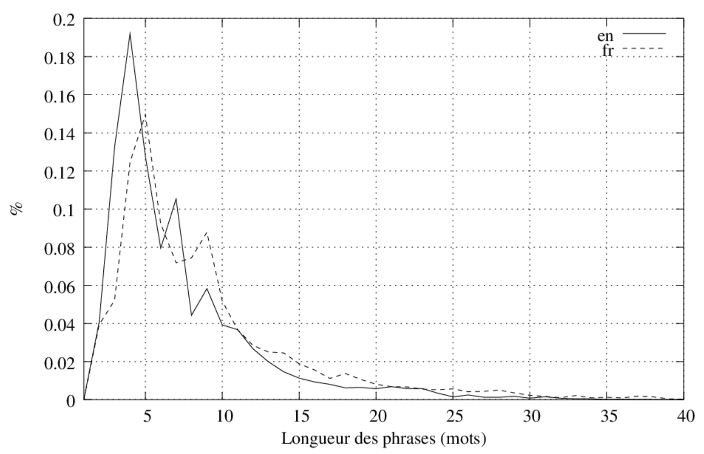


FIGURE 1.3: Distribution des phrases en nombre de mots.

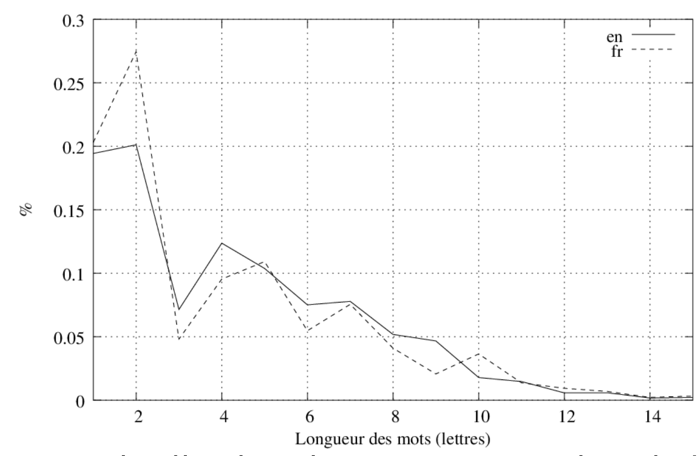


FIGURE 1.4: Distribution des mots en nombre de lettres.

Une autre propriété intéressante de notre bitexte est que seulement 8,6% de ses phrases n'y apparaissent qu'une seule fois. Inversement, certaines phrases telles que celles utilisées pour annoncer les températures maximales et minimales d'une journée apparaissent plusieurs centaines de milliers de fois dans le bitexte. Si les phrases rencontrées dans les bulletins météorologiques sont souvent les mêmes, nous pouvons nous

demander dans quelle mesure nous pourrions traduire les bulletins simplement en consultant une mémoire des traductions observées dans notre bitexte d'entraînement TRAIN. Nous allons donc maintenant étudier les possibilités et les limites d'une telle approche.

Chapitre 2
Approche basée sur une mémoire de traduction

Au chapitre 1, nous avons montré comment nous avons construit un très grand bitexte de phrases françaises et anglaises issues des données météorologiques qui nous ont été fournies par le CMC. Nous en avons étudié les propriétés linguistiques et nous avons noté un très fort taux de répétition des phrases. Cela nous a suggéré d'essayer une approche par mémoire de traduction.

Dans ce chapitre, nous décrivons comment nous avons construit une mémoire de traduction. Nous expliquons également comment nous nous en servons pour la traduction de bulletins météorologiques et nous évaluons les performances de notre technique. Nous montrons que cette approche simple amène des résultats très satisfaisants. Nous en identifions tout de même les limites et nous étudierons des alternatives aux chapitres 3 et 4.

2.1 Principe d'une mémoire de traduction

L'utilisation d'une archive de traductions pour la traduction automatique a été proposée par Peter Arthern en 1979 [9]. Cette technique, connue depuis sous le nom de mémoire de traduction, est basé sur un principe intuitivement simple : si nous disposons d'une liste de phrases associées à leur traduction, nous pouvons traduire les nouvelles occurrences de ces phrases en les cherchant dans la liste et en réutilisant la même traduction. Notez que qu'il est fait l'hypothèse qu'une phrase donnée a toujours la même traduction. Cela n'est pas toujours exact en raison des nombreuses homonymies présentes dans les langues naturelles mais il est raisonnable de penser que cette hypothèse est tout de même correcte lorsque nous nous limitons à des textes portant sur un même sujet comme la météorologie.

Une limitation connue des mémoires de traduction vient du fait que les répétitions de phrases entières sont très rares dans la plupart des textes. Langlais *et al.* [17] ont essayé de rechercher 1 260 phrases tirées d'articles du journal français “Le Monde” dans la base de données TSRali[1] et ils se sont ainsi apperçu que moins de 4% des phrases ont été trouvées dans la mémoire que représente la base de données. Les traducteurs qui travaillent sur des documents techniques ou tout autre

type de documents ayant un fort taux de répétition des phrases utilisent tout de même souvent des logiciels de de traduction assistée par ordinateur (TAO) incorporant une mémoire de traduction. Ainsi, lorsque le traducteur doit traduire une phrase qu'il a déjà traduite auparavant, le logiciel va automatiquement lui suggérer de réutiliser la même traduction. La mémoire du logiciel est basée sur l'historique des traductions produites par le traducteur. Certains logiciels peuvent également construire une mémoire *a priori* à partir d'un bitexte déjà produit, comme nous allons le faire nous-même. Parmi les logiciels de TAO les plus connus, on compte *Translation Manager/2* de la société IBM, *Déjà-Vu* de la société Atril et *Translator's Workbench* de la société Trados.

[1] Cette base de données de traductions contenant 15 années de débats parlementaires canadiens issus du corpus HANSARD est accessible par Internet sur le site `http://www.tsrali.com`.

Une solution pour augmenter l'efficacité des mémoires de traduction consiste à ne pas construire une mémoire de traduction au niveau des phrases mais plutôt à un niveau inférieur comme par exemple une mémoire de traduction des groupes de mots. Ces mémoires dites sous-phrastiques permettent de traduire les segments de phrases récurrents, ce qui facilite déjà la tâche du traducteur [26].

D'autres manières d'améliorer l'utilisation des mémoires phrastiques ont été proposées. Par exemple, à défaut de trouver la même phrase ou expression dans la mémoire, nous pouvons peut-être trouver une phrase suffisamment proche pour que sa traduction soit utile. Les travaux récents dans le domaine des mémoires de traduction phrastiques portent surtout sur les techniques d'appariement approximatif des phrases (*fuzzy matching*). Ces approches exploitent différentes techniques d'analyse morphologique, d'analyse de surface (*chunking*) ou encore d'identification et de traitement modulaire des entités nommées et des expressions numériques [22].

Dans le cas de la traduction des bulletins météorologiques, nous avons déjà montré que nous avons un taux élevé de répétitions de phrases entières. Nous pouvons donc espérer qu'une mémoire de traduction phrastique nous donnera de bons résultats. Pour nous en assurer, nous avons calculé que 83% des phrases du corpus BLANC se trouvent dans le corpus TRAIN. Autrement dit, si notre système devait traduire BLANC en disposant d'une mémoire des traductions de toutes les phrases de TRAIN, il ne lui resterait plus à traduire que les 17% de phrases qui ne sont pas simplement disponibles dans la mémoire. Convaincus de l'efficacité d'une mémoire de traduction phrastique pour la traduction des bulletins météorologiques, nous avons construit une telle mémoire.

2.2 Construction de la mémoire

Le premier paramètre à déterminer pour la construction de notre mémoire de traduction est le nombre de phrases sources qu'elle contiendra. Regardons comment varie la couverture de la mémoire (le pourcentage de phrases à traduire trouvées telle quelle dans la mémoire) en fonction du nombre de phrases sources anglaises qu'elle contient. Ce paramètre est présenté par la courbe de la figure 2.1. La couverture de la mémoire augmente logarithmiquement avec la taille de la mémoire jusqu'à ce que celle-ci atteigne la taille critique d'environ 10 000 phrases avec une couverture de près de 78%. Au-delà, la croissance de la couverture commence à ralentir. Il est clair que nous obtiendrons la meilleure couverture en incorporant dans la mémoire toutes les phrases de TRAIN (ce qui représente 4 188 100 phrases et environ 70 Mo de données). Nous aurons alors une couverture de 83%. Mais si la compacité de la mémoire avait été un critère essentiel dans notre projet, nous aurions pu ne garder que les 18 872 phrases qui apparaissent au moins 10 fois dans TRAIN (seulement 4 Mo) tout en conservant une couverture de presque 80%. Une mémoire plus petite serait aussi plus rapide à parcourir et donc sa vitesse de traduction serait plus grande. Comme nous voulons avant tout favoriser la qualité des traductions, nous avons opté pour une mémoire complète de TRAIN.

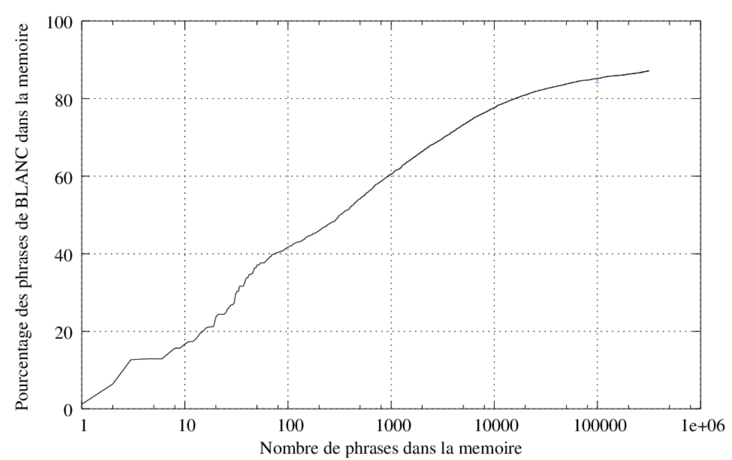


FIGURE 2.1: Couverture sur BLANC de la mémoire en fonction de sa taille.

Une manière d'augmenter la couverture de notre mémoire tout en réduisant sa taille consiste à factoriser certains invariants de traduction comme les données chiffrées (les heures, les dates ou les températures par exemple). Prenons les phrases utilisées pour annoncer des températures maximales négatives et leurs traductions :

- MAX NEAR MINUS 1 ↔ MAX PRES DE MOINS 1
- MAX NEAR MINUS 2 ↔ MAX PRES DE MOINS 2
- MAX NEAR MINUS 3 ↔ MAX PRES DE MOINS 3
- ...

Au lieu de mémoriser chacune de ces paires, nous pourrions simplement mémoriser un patron comme :

- MAX NEAR MINUS __INT__ ↔ MAX PRES DE MOINS __INT__

Par la suite, supposons que nous voulons traduire la phrase MAX NEAR MINUS 5. Nous commençons par la transformer en MAX NEAR MINUS __INT__ (en nous rappelant que __INT__ remplace 5). La mémoire nous donne alors la traduction MAX PRES DE MOINS __INT__. Il nous suffit de remplacer __INT__ par 5 pour obtenir la traduction finale : MAX PRES DE MOINS 5. Nous avons ainsi dans la mémoire une seule phrase pour traduire toutes les annonces de températures maximales négatives rencontrées dans TRAIN et cette phrase permettra même de traduire de nouvelles annonces de températures si le système est utilisé une année où les températures sont plus froides que celles rencontrées en 2002 et 2003.

Le même principe peut s'appliquer à d'autres classes de tokens qui sont toujours traduits d'une seule manière. Nous avons retenu les classes suivantes car elles sont particulièrement appropriées à notre corpus :

- __COORD__ pour les coordonnées cartographiques.
- __DAY__ pour les noms de jours.
- __INT__ pour les entiers.
- __MONTH__ pour les noms de mois.
- __PONCT__ pour les éléments de ponctuation.
- __PRCT__ pour les pourcentages.
- __RANGE__ pour les intervalles numériques.
- __TEL__ pour les numéros de téléphone.
- __TIME__ pour les heures.

D'autres classes de tokens nous avaient semblé intéressantes comme les noms propres (villes, régions, fleuves, lacs...) ou encore les directions cardinales (nord, sud, est, ouest...) mais malheureusement ces tokens peuvent parfois être ambigus. Par exemple, si nous remplaçons toutes les occurrences de EST par un marqueur désignant les points cardinaux, nous risquons de remplacer par erreur des occurrences du verbe être à la troisième personne du singulier du présent de l'indicatif. Il y a de

nombreux autres exemples de telles homographies, ce qui nous empêche d'utiliser toutes les classes que nous souhaiterions.

Pour commencer la construction de notre mémoire, nous avons créé une copie de notre corpus où toutes les occurrences des tokens appartenant aux classes de la liste ci-dessus ont été remplacées par les marqueurs correspondants. Un extrait de cette version marquée de notre corpus est présenté à la figure 2.2. Nous avons sérialisé les marqueurs afin de distinguer plusieurs marqueurs identiques dans une même phrase. Les numéros attribués à chaque marqueur sont simplement dans l'ordre croissant de la position du marqueur dans la phrase. Cela pourrait poser un problème si l'ordre des marqueurs venait à être inversé entre la phrase source et sa traduction mais cela ne semble pas se produire dans notre corpus.

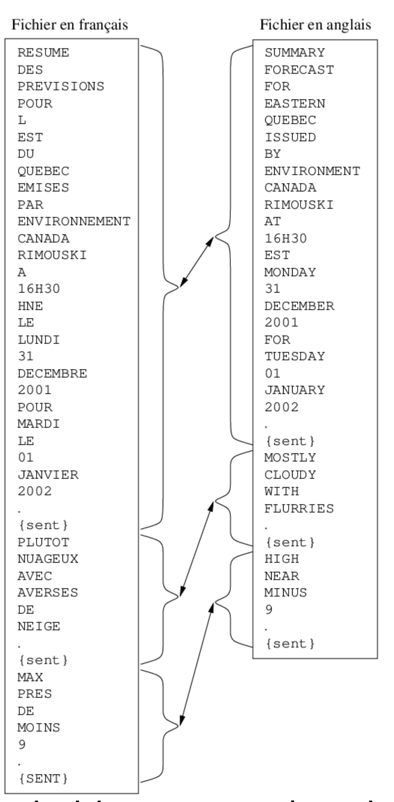


FIGURE 2.2: Extrait du bitexte avec les phrases tokenisées, alignées et marquées.

Notre nouveau corpus comporte 295 972 phrases marquées différentes, soit près de 40% de phrases en moins que sans le marquage. La taille de la mémoire est donc considérablement réduite par le marquage. De plus, nous avons maintenant 87% des phrases du corpus BLANC qui sont vues dans la mémoire. Nous avons donc amélioré la couverture précédente de 4%, ce qui n'est pas négligeable.

Comme certaines phrases ont plusieurs traductions différentes dans TRAIN, nous devons décider du nombre maximal de traductions que nous gardons dans la mémoire pour chaque phrase source. La figure 2.3 montre la distribution du nombre de traductions par phrase source dans TRAIN. 89% des phrases anglaises ont une seule traduction française et les phrases sources ayant plus de 3 traductions sont exceptionnelles. Nous avons donc limité notre mémoire à 5 traductions au plus par phrase source. Pour les 11% de phrases ayant plusieurs traductions, nous gardons les 5 traductions les plus fréquentes. La régularité des traductions indique clairement qu'elles ont été produites automatiquement. Les rares variations dans ces traductions viennent du fait qu'elles ont été vérifiées et, au besoin, corrigées par des traducteurs humains. Nous pouvons donc considérer que ces traductions sont d'une qualité jugée satisfaisante par des traducteurs professionnels.

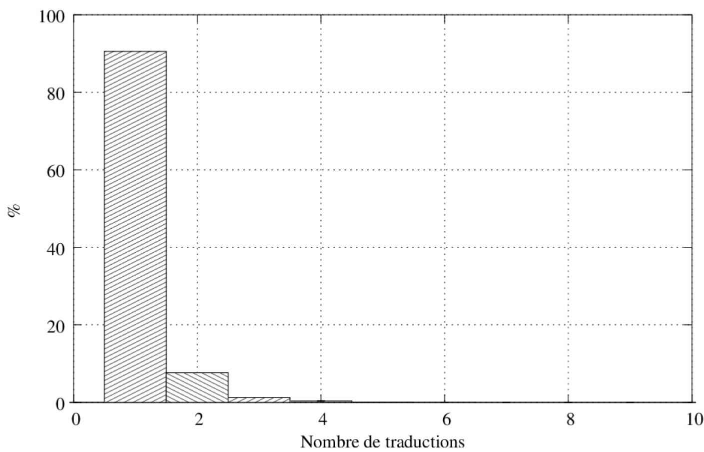


FIGURE 2.3: Distribution du nombre de traductions par phrase source dans TRAIN.

Une fois les paramètres de notre mémoire choisis, nous avons construit notre mémoire dans laquelle nous pouvons retrouver la traduction d'une phrase donnée. Les marqueurs de classe nous permettent de trouver dans la mémoire des phrases qui n'ont pas été vues telles quelles dans le corpus d'entraînement. Il s'agit en quelque sorte d'une première technique d'appariement approximatif. Toutefois nous souhaiterions avoir également une technique d'appariement approximatif qui ne se limite pas à certaines classes de tokens.

2.3 Appariement approximatif

Pour formaliser notre description de la mémoire, nous pouvons la définir comme un ensemble de groupes de traductions :

$$M = (T_1, T_2, T_3, \ldots, T_n) \tag{2.1}$$

où n est le nombre de phrases sources contenues dans la mémoire. Un groupe de traductions T_i est constitué de la manière suivante :

$$T_i = (e_i, (f_{i1}, n_{i1}), (f_{i2}, n_{i2}), (f_{i3}, n_{i3}), \ldots, (f_{ik_i}, n_{ik_i})) \quad (2.2)$$

où e_i est la i-ème phrase source (anglaise) dans la mémoire et chaque f_{ij} est une phrase cible (française) observée n_{ij} fois dans le corpus d'entraînement comme traduction de e_i. k_i est le nombre de traductions gardées dans la mémoire pour la phrase e_i. Nous avons fixé un maximum de $m = 5$ traductions par source donc $1 \leq k_i \leq 5$. Le nombre d'occurrences N_i d'une phrase source e_i est donnée par la formule :

$$N_i = \sum_{j=1}^{k_i} n_{ij} \quad (2.3)$$

La distance d'édition entre les phrases A et B est donc de 2 opérations (1 Remplacement, 0 Insertion et 1 Suppression).

Phrase A	AVERSES	DE	NEIGE	FONDANTE	CE	SOIR	.
Phrase B	AVERSES	DE	PLUIE		CE	SOIR	.
Opérations			R	S			

Figure 2.4: Calcul d'une distance d'édition.

Si, même après le marquage, la phrase à traduire ne se trouve pas dans la mémoire, nous voudrions retrouver dans la mémoire les phrases les plus proches en espérant que leurs traductions seront satisfaisantes. Les techniques d'appariement approximatif évoquées au début de ce chapitre utilisent différentes définitions de la proximité entre deux phrases. Une définition classique est la distance d'édition de Levenshtein : la distance entre deux phrases est le nombre minimum de mots qu'il faut insérer, effacer ou remplacer pour transformer une phrase en l'autre. La figure 2.4 montre un exemple de calcul de distance d'édition entre deux phrases. Un des principaux avantages de la distance de Levenshtein est qu'il existe un algorithme de programmation dynamique permettant de la calculer rapidement.

Notre technique d'appariement approximatif est la suivante. Pour une phrase e à traduire, nous recherchons dans la mémoire les 10 phrases sources e_i les plus proches de e en terme de distance d'édition. Pour trouver ces phrases e_i, nous devons parcourir la mémoire et calculer la distance d'édition de e avec chaque phrase source e_i. Parmi les phrases à distance d'édition égales, nous gardons en priorité celles qui sont les plus fréquentes dans le corpus d'entraînement. Eventuellement, si la

phrase *e* se trouve telle quelle dans la mémoire, nous aurons $e_1 = e$. Les autres phrases e_i sélectionnées seront à une distance d'édition supérieure ou égale à 1 par rapport à *e*.

Pour chaque e_i sélectionnée, la mémoire possède un groupe de traductions T_i. Supposons sans perte de généralité que la mémoire sélectionne les groupes T_1 à T_{10}. Dans chacun de ces groupes, nous avons de 1 à 5 traductions candidates. Nous nous retrouvons donc avec 10 à 50 traductions françaises possibles. Nous voulons maintenant donner un score à chaque traduction qui favorise celles provenant des phrases les plus proches de la phrase à traduire. A distance égale, nous avons préférons encore une fois les traductions les plus fréquentes dans le corpus d'entraînement.

Pour calculer le score s_{ij} de la traduction f_{ij} du groupe T_i, nous utilisons la formule suivante :

$$s_{ij} = -d(e_i, e) - \left(1 - \frac{\sum_{\substack{d(e_p,e)=d(e_i,e) \\ f_{pq}=f_{ij}}} n_{pq}}{\sum_{d(e_p,e)=d(e_i,e)} N_p} \right) \qquad (2.4)$$

L'idée de cette formule de score est que la partie entière du score dépend de la distance d'édition et la partie fractionnaire dépend de la fréquence de la traduction relativement aux autres traductions rencontrées pour la même phrase source. Comme une même traduction peut apparaître dans plusieurs groupes T_i, nous ne gardons que le meilleur score pour chacune (le score pour le groupe T_i pour lequel la distance $d(e_i,e)$ est la plus petite). Nous connaissons le cas le plus favorable pour notre mémoire de traduction (la phrase à traduire se trouve telle quelle dans la mémoire et n'a qu'une seule traduction connue) mais pas le cas le plus défavorable. Il nous faut une échelle de score bornée à une seule extrémité. Nous avons donc choisi de fixer le meilleur score à zéro et les scores moins bons sont les scores inférieurs. C'est la raison pour laquelle nos scores sont négatifs. Une fois toutes les traductions classées par score décroissant, nous pouvons ne garder par exemple que les 10 premières du classement.

2.4 Performances de la mémoire

Nous avons implanté la recherche de traduction dans la mémoire avec appariement approximatif tel que nous l'avons décrit précédemment. Pour des raisons d'efficacité, nous avons cette fois utilisé le langage de programmation C++.

Pour évaluer les performaces de notre mémoire, nous allons utiliser les métriques traditionnelles dans le domaine de la traduction automatique. La mesure de *Word Error Rate* (WER) est la distance d'édition normalisée moyenne entre chaque traduction candidate et la référence. Si la référence est la phrase `AVERSES DE NEIGE FONDANTE CE SOIR` et la traduction candidate est la phrase `AVERSES DE PLUIE CE SOIR`, la distance d'édition entre ces deux phrases est 2 (remplacer `NEIGE` par `PLUIE` et supprimer `FONDANTE`). Nous savons par ailleurs que la distance d'édition maximale entre deux phrases est égale à la longueur de la plus grande des deux phrases[2]. Si nous normalisons la distance 2 par la longueur 6, nous obtenons un WER de 33% pour cet exemple. Pour calculer le WER de plusieurs traductions, nous faisons la moyenne du WER de chaque phrase. Le WER nous donne donc une estimation de la proportion de mots erronés dans chaque traduction proposée.

[2] Nous pouvons toujours passer de la phrase la plus longue à la phrase la plus courte en supprimant le nombre de mots en trop dans la phrase la plus longue puis en substituant tous les mots restants.

Le *Sentence Error Rate* (SER) est le pourcentage des traductions proposées qui ne sont pas parfaitement identiques aux traductions de référence. Si par exemple le SER est faible mais le WER est grand, cela veut dire que nous avons beaucoup d'erreurs au niveau des mots concentrées dans peu de phrases.

Le score BLEU proposé par Papineni *et al.* [24] est une mesure pour laquelle la traduction candidate et celle de référence sont comparées non seulement au niveau des mots mais également au niveau des bigrammes, trigrammes *et cetera*. Pour calculer le score BLEU entre une traduction candidate *c* et une traduction de référence *r*, nous avons la formule suivante :

$$\text{BLEU}(c,r) = BP \times e^{\sum_{n=1}^{N} \frac{|n\text{-grammes}_c \cap n\text{-grammes}_r|}{N \times |n\text{-grammes}_c|}} \quad (2.5)$$

où N est la taille maximale des n-grammes considérés (4 généralement) et n-grammes$_c$ et n-grammes$_r$ sont respectivement les ensembles de n-grammes des phrases c et r. BP (la *brevity penalty*) est définie comme suit :

$$BP = \min\left(1, e^{\frac{|c|}{|r|}}\right) \quad (2.6)$$

Le coefficient *BP* est là pour éviter que BLEU ne favorise les traductions candidates courtes pour lesquelles $|n\text{-grammes}_c|$ est petit, ce qui augmente artificiellement le quotient dans l'exponentielle de la formule de BLEU. Par contre, BLEU favorise les traductions candidates contenant les *n*-grammes les plus courants alors que ce sont rarement ceux qui sont les plus porteurs de sens. NIST est une variante de BLEU proposée par l'organisme du même nom qui essaye de valoriser la traduction correcte des *n*-grammes plus rares parce qu'ils sont *a priori* plus difficiles à traduire [4].

Tout comme WER et SER, BLEU est normalisé entre 0 et 1. Il s'exprime donc souvent en pourcentage. Par contre, NIST n'est pas normalisé. Nous pouvons cependant connaître le NIST maximal que nous pouvons obtenir sur une référence donnée en calculant le NIST de la référence sur elle-même. Nous obtenons ainsi un NIST maximal de 12,27 pour BLANC.

La table 2.1 donne les scores[3] de la traduction de BLANC par notre mémoire de traduction phrastique constituée avec TRAIN.

3 Les scores NIST et BLEU ont été calculés avec le programme mteval (version 11a) disponible à `http://www.nist.gov/speech/tests/mt/`.

WER	SER	NIST	BLEU
8,78%	23,92%	11,2726	87,04%

TABLE 2.1: Scores de la mémoire de traduction.

Les scores que nous obtenons sont déjà plus élevés que ceux qu'obtiennent généralement les systèmes de traduction état-de-l'art sur des corpus standards. A titre d'exemple, les meilleurs systèmes actuels appliqués au corpus Hansard des discours parlementaires canadiens obtiennent généralement un WER autour de 60% et un SER entre 80 et 90% [6]. Comme nous l'espérions, la simplicité et la répétitivité de notre corpus météorologique font qu'une telle approche donne déjà des scores très satisfaisants.

Nous sommes tout de même un peu surpris que le SER ne soit pas encore plus bas. En effet, nous avons dit que 87% des phrases de BLANC sont dans la mémoire. Nous serions donc en droit de nous attendre à avoir au moins 87% des traductions parfaitement correctes. Malheureusement, il arrive que la référence ne soit pas cohérente avec elle-même : dans 7,24% des cas, la traduction de référence pour une phrase source donnée n'est pas la traduction la plus couramment rencontrée pour cette même phrase source dans le reste de la référence ! Nous avons donc environ 7% de SER qui sont inévitables.

Il ne faut pas non plus oublier les limitations de ces scores : la mémoire peut facilement traduire une phrase par son contraire tout en gardant un bon score WER car il suffit généralement d'ajouter une négation ou de remplacer un mot par son contraire dans une phrase pour en changer complètement le sens ! Nous allons examiner le type d'erreurs commises par la mémoire.

2.5 Analyse d'erreurs

Comme nous l'avons vu précédemment, nous avons un peu moins d'un quart des traductions issues de la mémoire qui ne sont pas identiques aux traductions de référence. Pour illustrer le comportement de la mémoire, nous avons sélectionné 3 exemples de traductions : un exemple où la phrase à traduire se trouve telle quelle dans la mémoire (fig. 2.5) ; deux exemples où la phrase dans la mémoire la plus proche de celle à traduire est à distance de 2 (fig. 2.6 et 2.7) et un exemple où la phrase la plus proche est à une distance de 5 (fig. 2.8).

Source :
THE NEXT SCHEDULED FORECAST WILL BE ISSUED AT 11H00 .
Source avec marqueurs :
THE NEXT SCHEDULED FORECAST WILL BE ISSUED AT __TIME1__ __PONCT1__
Mémoire (distance = 0) :
THE NEXT SCHEDULED FORECAST WILL BE ISSUED AT __TIME1__ __PONCT1__
Cible avec marqueurs :
PROCHAINES PREVISIONS EMISES A __TIME1__ __PONCT1__
Cible sans marqueurs :
PROCHAINES PREVISIONS EMISES A 11H00 .
Référence :
PROCHAINES PREVISIONS EMISES A 11H00 .

Figure 2.5: Traduction issue de la mémoire à distance 0.

Source :
TODAY .. PERIODS OF SNOW AMOUNT 10 CM .
Source avec marqueurs :
TODAY __PONCT1__ PERIODS OF SNOW AMOUNT __INT1__ CM __PONCT2__
Mémoire (distance = 2) :
TODAY __PONCT1__ PERIODS OF SNOW GIVING UP TO __INT1__ CM __PONCT2__
Cible avec marqueurs :

AUJOURD HUI __PONCT1__ NEIGE PASSAGERE DONNANT PRES DE __INT1__ CM __PONCT2__
Cible sans marqueurs :
AUJOURD HUI .. NEIGE PASSAGERE DONNANT PRES DE 10 CM .
Référence :
AUJOURD HUI .. NEIGE PASSAGERE ACCUMULATION DE 10 CM .

Figure 2.6: Traduction issue de la mémoire à distance 2.

Source :
DRIFTING SNOW AT TIMES LOCALLY .
Source avec marqueurs :
DRIFTING SNOW AT TIMES LOCALLY __PONCT1__
Mémoire (distance = 2) :
SNOW AT TIMES HEAVY __PONCT1__
Cible avec marqueurs :
NEIGE PARFOIS FORTE __PONCT1__
Cible sans marqueurs :
NEIGE PARFOIS FORTE .
Référence :
POUDRERIE BASSE PASSAGERE PAR ENDROITS .

Figure 2.7: Autre traduction issue de la mémoire à distance 2.

Source :
BLIZZARD CONDITIONS WILL BEGIN LATER THAN PREVIOUSLY FORECAST .
Source avec marqueurs :
BLIZZARD CONDITIONS WILL BEGIN LATER THAN PREVIOUSLY FORECAST __PONCT1__
Mémoire (distance = 5) :
SNOWFALL AMOUNTS ARE HIGHER THAN PREVIOUSLY FORECAST __PONCT1__
Cible avec marqueurs :
ACCUMULATION DE NEIGE SUPERIEURE A CE QUI ETAIT PREVU __PONCT1__
Cible sans narqueurs :
ACCUMULATION DE NEIGE SUPERIEURE A CE QUI ETAIT PREVU .
Référence :
LE BLIZZARD DEBUTERA PLUS TARD QUE PREVU .

Figure 2.8: Traduction issue de la mémoire à distance 5.

Dans le premier exemple, la phrase source étant dans la mémoire, la bonne traduction est directement extraite de la mémoire. Dans le second

exemple, la meilleure phrase dans la mémoire est à une distance de 2 de celle recherchée mais elle a un sens similaire donc la traduction est correcte (même si elle n'est pas identique à la référence). Par contre, dans le troisième exemple, pour une même distance de 2, la mémoire trouve parfois des phrases dont le sens est différent de celui de la phrase à traduire et la traduction est insatisfaisante. Et quand la distance devient trop grande, comme pour le quatrième exemple, la traduction extraite de la mémoire est presque toujours complètement erronée.

Pour déterminer globalement les tendances d'erreurs dans nos traductions, nous avons aligné mot à mot chaque traduction erronée avec la traduction de référence correspondante. Pour faire cet alignement, nous avons utilisé notre algorithme de calcul de distance d'édition. Cette fois-ci, ce n'est pas la distance elle-même qui nous intéresse mais plutôt la séquence d'opérations d'édition utilisée pour la calculer. Reprenons l'exemple de la figure 2.6. Si nous calculons la distance d'édition entre la référence et la cible proposée, nous obtenons une distance de 2 et par remplacement de `ACCUMULATION` par `DONNANT` suivi de l'insertion de PRES. Ce groupe d'opérations d'édition vient du fait que la référence attend `ACCUMULATION` là où notre cible propose `DONNANT` PRES. En repérant les groupes d'opérations d'édition consécutives, nous trouvons donc les groupes de mots erronés et leurs corrections selon la référence.

En utilisant cette technique, nous avons tenté d'isoler tous les mots ou groupes de mots incorrects dans nos traductions de BLANC et nous leur avons associé leurs corrections. Nous avons 6 740 erreurs différentes que nous avons triées par nombre d'occurrences décroissant.

Nous donnons en table 2.2 les 20 erreurs les plus fréquentes avec leur nombre d'occurrences. Nous remarquons en premier que les erreurs aux lignes 1, 2 et 5 de la table n'en sont en fait qu'une seule : parfois la traduction issue de la mémoire contient une expression telle que `POSSIBILITE DE 30 POURCENT` alors que la référence propose `PROBABILITE DE 30 POURCENT` ou `30 POURCENT DE PROBABILITE`. C'est la différence la plus fréquente entre nos traductions et la référence et il ne s'agit pas véritablement d'une erreur car les expressions substituées ont le même sens. D'autres différences dans la table ne sont pas des erreurs mais plutôt des synonymes comme `STABLES` et `STATIONNAIRES`, `MAX` et `MAXIMUM`, `LE MATIN` et `EN MATINEE`...

Nb.	Référence	Candidat
6119		POSSIBILITE DE

5556	DE PROBABILITE	
4917		DE
1902	DE	A
526	PROBABILITE DE	
355	STABLES	STATIONNAIRES A
351	TOT	
348	COUP	COUPS
323	TARD	
263	GRADUEL	
260	A	DE
250	LE MATIN	EN MATINEE
239	LES	
233	NEIGE	PLUIE
222	SERONT	
214	,	.
213	HNP	HAP
207		VENTS DE
199	MAX	MAXIMUM DE
189	LE	

Table 2.2: Les erreurs les plus fréquentes de la mémoire.

La suppression d'adjectifs comme GRADUEL ou d'adverbes comme TOT est aussi une erreur assez peu importante. Le sens d'une phrase comme AVERSES DE PLUIE TOT CE MATIN ne repose pas vraiment sur l'adverbe TOT. La même remarque est valable pour erreurs sur les articles (lignes 2, 3, 10...) ou les fautes d'accord (ligne 8) : elles rendent les phrases incorrectes du point de vue grammatical mais le sens de la phrase est préservé.

Par contre le fait de substituer NEIGE par PLUIE (ligne 14) est une erreur importante car cela change complètement l'information que fera passer le bulletin.

Notre conclusion de cette analyse est qu'il faut rester prudent face aux scores obtenus par la mémoire seule et se convaincre que cette approche, bien que très utile dans 87% des cas, ne suffit pas pour faire un véritable système de traduction fiable. Nous pourrions ajouter des règles de post-traitement pour corriger les erreurs connues les plus fréquentes mais cela ne nous servirait pas au long terme quand de nouvelles erreurs surviendront. Nous allons donc plutôt voir comment nous pouvons pousser plus loin notre exploitation de la mémoire pour

corriger automatiquement la traduction des 13% de phrases qui ne sont pas telles quelles dans la mémoire.

Chapitre 3
Consensus par l’alignement multiple de phrases

Au chapitre précédent, nous avons montré que près de 9 fois sur 10, notre mémoire de traduction contient la phrase exacte que nous voulons traduire. Nous avons vu cependant que si la phrase la plus proche trouvée dans la mémoire a une distance d’édition de seulement une ou deux opérations d’édition, nous pouvons espérer que la différence de sens entre entre la phrase trouvée et celle que nous voulons traduire soit minime. Nous avons vu également qu’une simple substitution peut changer le sens d’une phrase. Quand la distance est faible mais non nulle, nous ne pouvons donc pas vraiment prédire la qualité de nos traductions. Par contre, quand la distance d’édition devient plus grande, nous savons que nous ne pouvons plus nous contenter de garder les traductions fournies par la mémoire car celles-ci contiennent probablement des groupes de mots entiers incorrects.

Dans ce chapitre, nous allons montrer que lorsque la mémoire contient des phrases proches de la phrase source, les différentes traductions proposées par la mémoire peuvent être comparées et combinées automatiquement pour obtenir des traductions de meilleure qualité. Nous verrons que cette technique ne peut toutefois pas s’appliquer quand les différentes traductions proposées par la mémoire sont toutes trop erronées (typiquement quand la distance d’édition minimale est trop grande, c’est-à-dire quand la phrase à traduire ne ressemble à rien de ce qui est dans la mémoire). Pour ces phrases nouvelles, nous proposerons une solution au chapitre suivant.

3.1 Multiples traductions de la mémoire

La mémoire nous fournit les traductions de plusieurs phrases proches de celle que nous voulons traduire. Les traductions sont classées en ordre décroissant de score. Prenons par exemple la sortie de la mémoire présentée à la figure 3.1. Les phrases cibles sont les traductions que nous donne la mémoire. La plupart des traductions cibles s'accordent sur le fait que le début de la phrase est `MAXIMUM DE`. De même, la plupart des traductions cibles se terminent par `CE MATIN`, parfois précédé de `TOT` mais pas toujours. Par contre certaines traductions comme `BRUMEUX TOT CE MATIN` n'ont aucun rapport sémantique avec la traduction recherchée même si leur distance d'édition est relativement faible. Nous voudrions donc faire une sorte de consensus entre les traductions : nous débarrasser des traductions trop éloignées des autres et construire à la place de nouvelles traductions composées de différentes combinaisons des parties les plus courantes des phrases proches les unes des autres.

Source : `HIGH 12 EARLY THIS MORNING .`
Référence : `MAXIMUM 12 TOT CE MATIN .`
Cible 0 : [-1.14286] `MAXIMUM DE 12 CE MATIN .`
Cible 1 : [-2.00000] `MAXIMUM 12 ATTEINT CE MATIN .`
Cible 2 : [-2.00000] `MAXIMUM DE 12 TOT CET APRES-MIDI .`
Cible 3 : [-2.00000] `MAXIMUM DE PLUS 12 TOT CE MATIN .`
Cible 4 : [-2.00155] `NAPPES DE BROUILLARD TOT CE MATIN .`
Cible 5 : [-2.00239] `BRUMEUX PAR ENDROITS TOT CE MATIN .`
Cible 6 : [-2.02381] `MAXIMUM DE 12 EN MATINEE .`
Cible 7 : [-2.02564] `BRUMEUX TOT CE MATIN .`
Cible 8 : [-2.04167] `MAXIMUM DE PLUS 12 CE MATIN`

.
Cible 9 : [-2.04348] MAXIMUM DE MOINS 12 CE MATIN .

FIGURE 3.1: Multiples traductions proposées par la mémoire.

3.2 Génération de traductions par consensus

Un article de Srinivas BANGALORE *et al.* [1] suggère d'améliorer un ensemble de traductions données par différents systèmes de traduction en alignant toutes les traductions d'une phrase source. L'alignement multiple ainsi obtenu permet de construire un automate fini qui combine les différentes parties de ces traductions. Ensuite, en pondérant les transitions de l'automate et en cherchant les meilleurs chemins qui en résultent, nous pouvons produire de nouvelles traductions qui font en quelque sorte le consensus de celles proposées par les différents systèmes. Comme les systèmes ne font pas tous la même erreur au même endroit, l'article montre ainsi que l'on peut obtenir des traductions meilleures que celles fournies individuellement par chaque système.

Système	WER
MT 1	70,2%
MT 2	76,3%
MT 3	64,8%
MT 4	53,1%
MT 5	50,3%
Consensus	49,0%

TABLE 3.1: Scores de WER obtenu par le consensus de BANGALORE *et al.*

Le tableau 3.1 montre les WER obtenus par BANGALORE avec le consensus de 5 systèmes anonymes utilisés pour traduire

des phrases tirées d'un bitexte de discussions en espagnol et en anglais. Les scores des systèmes pris individuellement ont été mis pour comparaison. Le gain obtenu par le consensus n'est pas négligeable.

Nous avons donc décidé d'adapter cette technique pour la sortie de notre mémoire de traduction lorsque la phrase à traduire n'a pas été trouvée telle quelle dans la mémoire (c'est-à-dire les phrases nouvelles). Par la suite, nous ne nous intéresserons plus aux phrases qui se trouvent dans la mémoire car nous considérons ces phrases comme un problème résolu par la mémoire.

3.2.1 Alignement multiple de traductions

Nous avons montré à la section 2.5 comment un alignement au niveau des mots entre deux phrases nous permet de trouver les parties communes de ces deux phrases. Nous avons dit que pour calculer une traduction consensus par la technique proposée par Bangalore, nous devons tout d'abord aligner entre elles toutes les phrases cibles données par la mémoire pour une même source.

L'alignement entre plus de deux phrases peut être vu comme une extension de celui entre deux phrases. Dans l'alignement de paires de phrases, nous obtenons un profil des deux phrases : la séquence des insertions, suppressions et remplacements nécessaires pour transformer une phrase en l'autre. La complexité algorithmique du calcul d'un profil est $O(L^2)$ où L est le nombre de tokens de la plus longue phrase. L'extension de cet algorithme pour aligner ensemble N phrases mène à une complexité exponentielle $O(L^N)$. Dans notre cas, nous avons $N = 10$ car la mémoire nous donne les 10 meilleures traductions trouvées. Nous avons donc besoin d'un algorithme plus rapide que $O(L^N)$.

Un problème similaire se pose lors du séquençage de chaînes de protéines (typiquement de l'ADN). En effet, les techniques de laboratoire actuelles permettent d'obtenir la séquence de brins d'ADN allant jusqu'à environ 500 protéines. Toutefois, ces procédés ne sont pas exempts d'erreurs. C'est la raison pour laquelle les généticiens commencent par dupliquer le brin qu'ils souhaitent séquencer puis ils déterminent expérimentalement la séquence de chaque copie et enfin ils utilisent un programme pour déterminer la séquence réelle en faisant un consensus entre les séquences des différentes copies. En raison de la taille des séquences à aligner, les généticiens ont eux aussi besoin d'un algorithme plus rapide que $O(N^L)$. Le plus souvent, ils utilisent l'algorithme heuristique de Feng-Doolittle connu sous le nom d'alignement multiple progressif [5]. L'algorithme se déroule ainsi :

1. Calculer la distance d'édition et le profil pour chacune des $\frac{N(N-1)}{2}$ paires de phrases.
2. Répéter les étapes suivantes jusqu'à ce qu'il ne reste qu'un seul profil :
   a. Sélectionner le profil pour la plus petite distance d'édition phrase à phrase, phrase à profil ou profil à profil.
   b. Calculer la distance d'édition entre ce profil et les autres phrases et profils.

Le résultat de l'algorithme est une structure d'arbre où les phrases sont les feuilles et les profils sont les autres noeuds de l'arbre. Plus deux phrases sont semblables, plus elle seront proches dans l'arbre. Notons que cet algorithme glouton ne trouve pas toujours le meilleur alignement multiple possible.

Plutôt que d'implanté par nous-même l'alignement multiple progressif, nous avons préféré utiliser les sources du

programme CLUSTAL W version 1.83 disponibles sur le site Internet du CEPBA-IBM Research Institute[1]. THOMPSON *et al.* [27] décrit en détails l'algorithme d'alignement multiple utilisée par CLUSTAL W qui est une version optimisée de l'alignement multiple progressif.

[1] ftp ://kaa.cepba.upc.es/cela_pblade/clustalw.tar.gz

Nous avons modifié CLUSTAL W de manière à l'adapter à nos besoins. Tout d'abord, nous avons fait en sorte que le programme n'accepte plus seulement les 23 lettres généralement utilisées pour représenter des protéines mais plutôt les 62 caractères alphanumériques (26 majuscules, 26 minuscules et 10 chiffres) qui nous servirons chacun à représenter un mot différent. Ensuite, nous avons aussi remplacé les matrices 23 × 23 représentant les affinités chimiques entre les différentes protéines par une matrice identité 62 × 62. Enfin nous avons désactivé les pénalités que le programme ajoute lorsque qu'un alignement nécessite d'introduire une série d'insertions ou une série de suppressions. De telles séries sont appelées des *gaps* par les généticiens et elles ne sont parfois pas souhaitables dans le séquençage de protéines alors que nous verrons plus tard qu'elles ne nous dérangent pas dans notre application.

Pour pouvoir utiliser CLUSTAL W, nous devons remplacer chaque mot du vocabulaire des traductions sorties de la mémoire par un caractère alphanumérique. Ensuite les différentes séquences de caractères obtenues sont données au programme qui nous donne en sortie les versions alignées de ces séquences avec des *gaps* insérés. En restituant les mots à la place des caractères (plus des mots vides pour chaque *gap*), nous obtenons des alignements de phrases. La figure 3.2 montre le déroulement de ces opérations sur les traductions prises en exemple dans la figure 3.1.

```
MAXIMUM DE 12 CE MATIN .
```

```
MAXIMUM 12 ATTEINT CE MATIN .
MAXIMUM DE 12 TOT CET APRES-MIDI .
MAXIMUM DE PLUS 12 TOT CE MATIN .
NAPPES DE BROUILLARD TOT CE MATIN .
BRUMEUX PAR ENDROITS TOT CE MATIN .
MAXIMUM DE 12 EN MATINEE .
BRUMEUX TOT CE MATIN .
MAXIMUM DE PLUS 12 CE MATIN .
MAXIMUM DE MOINS 12 CE MATIN .
```

⇓ Pré-traitement

```
ABCDEF
ACGDEF
ABCHIJF
ABKCHDEF
LBMHDEF
NOPHDEF
ABCQRF
NHDEF
ABKCDEF
ABSCDEF
```

⇓ Alignement multiple par CLUSTAL W

```
AB-C-DEF
A--CGDEF
AB-CHIJF
ABKCHDEF
LBM-HDEF
NOP-HDEF
AB-C-QRF
N---HDEF
```

```
ABKC-DEF
ABSC-DEF
```

⇓ Post-traitement

```
MAXIMUM DE            12         CE  MATIN
     .
MAXIMUM               12 ATTEINT CE  MATIN
     .
MAXIMUM DE            12 TOT     CET APRES-
MIDI .
MAXIMUM DE  PLUS      12 TOT     CE  MATIN
     .
NAPPES  DE  BROUILLARD   TOT     CE  MATIN
     .
BRUMEUX PAR ENDROITS     TOT     CE  MATIN
     .
MAXIMUM DE            12         EN  MATINE
E    .
BRUMEUX                  TOT     CE  MATIN
     .
MAXIMUM DE  PLUS      12         CE  MATIN
     .
MAXIMUM DE  MOINS     12         CE  MATIN
     .
```

Figure 3.2: Processus d'alignement multiple.

3.2.2 Transformation des alignements en automate

Une fois que nous avons obtenu un alignement de toutes les traductions tirées de la mémoire, une première solution pour obtenir une traduction consensus est de faire un vote de majorité : garder le mot le plus fréquent de chaque 'colonne' (chaque vide étant considéré comme un mot à part entière).

Toutefois, cette approche simpliste ne donne pas de bons résultats.

Une seconde technique plus intéressante consiste à transformer les phrases alignées en un automate fini déterministe pondéré . L'idée est de considérer une phrase comme une suite de transitions entre des états. Pour lire une phrase, nous commençons dans l'état de début de phrase. Puis, en lisant le premier mot, nous arrivons dans un premier état. Et ainsi de suite jusqu'à ce que nous arrivions dans l'état de fin de phrase. Nous pouvons donc voir une phrase comme une série de transitions (les mots) entre des états. Une phrase forme un simple automate linéaire mais si nous avons plusieurs phrases qui ont probablement des groupes de mots en commun, nous pouvons construire un seul automate pour toutes ces phrases. Prenons par exemple les traductions produites à la fin de la figure 3.2. La première traduction, par le procédé que nous venons d'expliquer, peut être vue comme l'automate de la figure 3.3. Ensuite nous ajoutons la seconde traduction en ajoutant des états et des transitions là où la seconde traduction diffère de la première. Le résultat est l'automate de la figure 3.4. Les transitions de l'automate sont pondérées par leur nombre d'occurrences. Si nous ajoutons maintenant la troisième traduction, nous obtenons l'automate de la figure 3.4. En continuant ce processus, nous arrivons finalement à l'automate complet représenté à la figure 3.6.

FIGURE 3.3: Automate pour la première traduction.

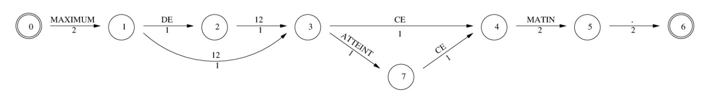


FIGURE 3.4: Automate pour les deux premières traductions.

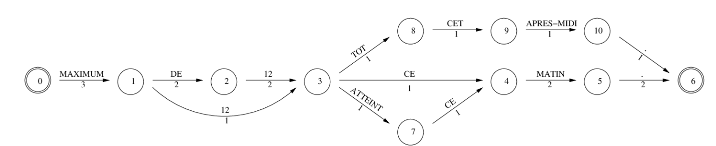


FIGURE 3.5: Automate pour les trois premières traductions.

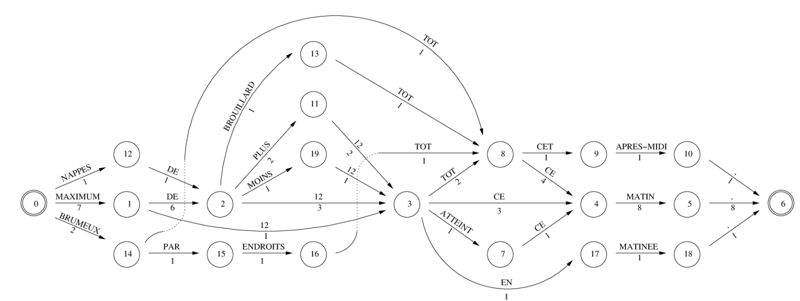


FIGURE 3.6: Automate pour les dix traductions de la figure 3.2.

Un programme peut facilement construire les automates à partir des alignements en suivant exactement le procédé que nous venons de décrire. Notre programme doit s'assurer que toutes les transitions qui mènent à un même état sont étiquetées par le même mot. Réciproquement tous les mots identiques qui sont alignés ensembles doivent être représentés par des transitions aboutissant au même état. Ainsi notre automate ne reconnaît pas seulement les dix traductions à partir desquelles il a été construit, il peut également reconnaître de nouvelles phrases en passant par des états communs à plusieurs phrases. Nous pouvons ainsi passer d'une phrase à une autre pour trouver de nouvelles phrases telles que par exemple `MAXIMUM DE PLUS 12 TOT CE MATIN`.

Par construction, une autre propriété de nos automates est que pour chaque état non initial ou final, la somme des poids des transitions entrantes est égale à la somme de ceux des transitions sortantes.

3.2.3 Extraction des traductions consensus

Un tel automate peut ensuite être fourni à un programme qui cherche dans l'automate les *n* chemins de plus grand poids (*n* pouvant être choisi par l'utilisateur). Comme, par construction, le poids d'un arcs dépend du nombre d'occurrences de la transition qu'il représente, les chemins obtenus représentent bien des traduction consensus.

Nous avons utilisé le logiciel CARMEL [12] disponible gratuitement sur le site de ISI[2] pour trouver les meilleurs chemins dans nos automates. La figure 3.7 montre la sortie du programme pour l'automate de la figure 3.6 (les traductions sont triées par ordre de poids décroissant).

[2] `http://www.isi.edu/licensed-sw/carmel/`

Source : `HIGH 12 EARLY THIS MORNING .`
Référence : `MAXIMUM 12 TOT CE MATIN .`
Cible 0 : `MAXIMUM DE PLUS 12 CE MATIN .`
Cible 1 : `MAXIMUM DE 12 CE MATIN .`
Cible 2 : `MAXIMUM DE PLUS 12 TOT CE MATIN .`
Cible 3 : `MAXIMUM DE 12 TOT CE MATIN .`
Cible 4 : `MAXIMUM DE TOT CE MATIN .`
Cible 5 : `MAXIMUM DE ENDROITS TOT CE MATIN .`
Cible 6 : `MAXIMUM DE BROUILLARD TOT CE MATIN .`
Cible 7 : `MAXIMUM DE MOINS 12 CE MATIN .`
Cible 8 : `MAXIMUM DE PLUS 12 EN MATINEE .`
Cible 9 : `MAXIMUM DE PLUS 12 ATTEINT CE MATIN .`

FIGURE 3.7: Traductions proposées après consensus.

Les phrases obtenues sont plus consistantes entre elles que celles que nous avions en entrée. Les phrases trop différentes des autres (comme les cibles 2, 4, 5 et 7 de la figure 3.1) ont été remplacées par de nouvelles phrases plus proches de la traduction recherchée (comme les cibles 8 et 9 de la figure 3.7). Globalement, la qualité des traductions proposées est améliorée : dans l'ancienne sortie, la troisième

phrase était déjà fausse alors que dans la nouvelle sortie, les deux premières traductions sont satisfaisantes et les deux suivantes nous semblent même grammaticalement meilleures que la traduction de référence !

Nous avons voulu évaluer la qualité des traductions consensus sur les 13 010 phrases de notre corpus BLANC qui ne sont pas dans la mémoire. Les résultats en termes de WER, SER, NIST et BLEU sont indiqués dans la table 3.2. Les meilleurs scores sont indiqués en gras. Clairement le consensus entre les phrases améliore le nombre de phrases identiques à la référence. C'est un résultat très positif pour cette approche. En revanche, il semble que le consensus introduise d'autres erreurs qui font légèrement monter le taux d'erreur sur les mots. Les scores NIST et BLEU sont aussi légèrement moins bons après le consensus. Nous allons maintenant essayer de voir s'il est possible de remédier à la détérioration de ces taux.

cibles	WER	SER	NIST	BLEU
avant consensus	**22,69%**	94,82%	**9,7853**	**66,56%**
après consensus	24,62%	**87,37%**	9,6624	64,78%

TABLE 3.2: Scores du consensus pour les phrases hors-mémoire.

3.3 Améliorer la pondération de l'automate

3.3.1 Contribution selon le rang

Lors de la pondération de l'automate, toutes les phrases issues de la mémoire ont la même influence alors que les phrases que la mémoire retourne en premier sont probablement meilleures que celles en dernier car elles proviennent de phrases sources plus proches de celles que nous voulons traduire. Nous allons donc essayer de prendre

en compte cette information par une nouvelle pondération de l'automate.

Introduire le rang de la phrase comme paramètre dans la pondération de l'automate n'est pas très difficile. Jusqu'à maintenant, chaque transition dans l'automate était pondérée selon son nombre d'occurrences parmi les phrases cibles. Chaque occurrence comptait pour 1 dans le poids de la transition. Si nous avons 10 phrases cibles par exemple, une autre possibilité est de décider maintenant qu'une occurrence dans la première phrase va compter pour 10 alors qu'une occurrence dans la seconde phrase ne comptera que pour 9 et ainsi de suite jusqu'à une contribution de 1 pour les transitions dans la dernière phrase. Ainsi les transitions observées dans la première phrase compteront deux fois plus que celles observées dans la sixième et 10 fois plus que celles observées dans la dernière.

En fait, il y a de nombreuses méthodes de pondération similaires que nous pouvons essayer afin d'augmenter ou réduire la différence de contribution entre les premières et les dernières phrases. Nous venons de présenter une fonction linéaire mais nous pouvons utiliser toute la famille de fonctions de contribution c_i définie comme suit :

$$c_i(t) = (r_{max} - r_t)^i \quad (3.1)$$

où t est la transition considérée, r_t est le rang de la phrase où t apparaît et r_{max} est le rang de la dernière phrase (autrement dit c'est le nombre de traductions proposées par la mémoire). Ainsi, lorsque toutes les transitions ont une contribution de 1, nous utilisions en fait c_0. La fonction linéaire décrite plus haut est c_1 et nous avons également essayé les contributions c_2 à c_5. Avec c_5, une transition dans la première phrase (parmi 10 toujours) compte pour 100 000 alors qu'une transition dans la dernière phrase compte toujours pour 1. Quelque soit la fonction de contribution

choisie, le poids d'un arc correspondant à une transition *t* dans l'automate reste la somme de toutes les pondérations des différentes occurrences de *t* au même endroit dans les phrases cibles. Tous les poids des arcs sont toujours normalisés de sorte que la somme des poids des arcs qui quittent un état de l'automate reste 1.

Notons également que si un mot apparaît à plusieurs endroits dans une même phrase, il ne s'agira pas de la même transition dans l'automate donc les contributions ne seront pas dupliquées. De même, les mots identiques dans des phrases différentes ne contribuent au même arc que s'ils ont été alignés lors de l'alignement multiple.

La table 3.3 donne les scores avant tout consensus (sortie de la mémoire brute) suivis des scores des consensus selon les différentes pondérations. Rappelons que nous ne travaillons toujours que sur les phrases de BLANC qui n'ont jamais été vues dans TRAIN. Les nouvelles pondérations (c_1 à c_5) ont toutes des meilleures scores que la mémoire seule sans consensus ou même que la pondération constante c_0. Par contre, il est difficile d'identifier une pondération meilleure que les autres car cela dépend du score considéré. Nous pouvons seulement dire que la pondération linéaire c_1 a les meilleures performances globales.

contribution	WER	SER	NIST	BLEU
aucune	22,69%	94,82%	9,7853	66,56%
c_0	24,62%	87,37%	9,6624	64,78%
c_1	22,97%	**85,53%**	9,9314	66,86%
c_2	22,60%	86,99%	9,9314	67,21%
c_3	22,40%	88,98%	9,9707	67,31%
c_4	22,20%	89,46%	**9,9670**	**67,47%**
c_5	**22,17%**	91,21%	9,9312	67,43%

Table 3.3: Scores des consensus avec différentes pondérations par le rang.

3.3.2 Lissage avec un modèle de langue

Une autre fonction de contribution pourrait être la probabilité donnée par un modèle de langue à la transition. En effet, à l'état 2 de l'automate de la figure 3.6, les arcs entrants sont tous étiquetés par le mot `DE`. Ainsi, l'état 2 est celui dans lequel nous nous trouvons après avoir rencontré le mot `DE` en deuxième position dans une phrase cible. Maintenant, chaque arc sortant de l'état 2 a aussi une étiquette (`BROUILLARD`, `PLUS`, `MOINS` ou 12). L'arc 2 → 12 porte l'étiquette `BROUILLARD` et représente donc le fait de rencontrer le mot `DE` suivi de `BROUILLARD`. Par construction, chaque arc de l'automate représente donc une paire de mots ou bigramme. Un modèle de langue des bigrammes du français pourrait donc nous donner une probabilité pour le bigramme `DE BROUILLARD` et cette probabilité pourrait être utilisée comme poids pour l'arc 2 → 12. Nous verrons plus en détails le fonctionnement des modèles de langue au chapitre suivant mais, pour l'instant, il suffit de savoir qu'un modèle de langue est capable de donner, pour une langue donnée, la probabilité d'un mot sachant les mots qui le précèdent. Le modèle établit ses probabilités par observation d'un large corpus d'entraînement (TRAIN dans notre cas). En pondérant notre automate avec ces probabilités, nous espérons faire ressortir les chemins dans l'automate qui ressemblent le plus au langage météorologique vu dans TRAIN.

Nous avons implanté ce second mode de pondération avec un modèle de Jelinek-Mercer interpolé (cf. section 4.1.1) à l'aide des outils développés à l'Université de Montréal par George Foster. Les résultats que nous avons obtenus sont présentés à la table 3.4.

cibles	WER	SER	NIST	BLEU
avant consensus	**22,69%**	**94,82%**	**9,7853**	**66,56%**
après consensus	37,93%	99,68%	7,2500%	45,57%

TABLE 3.4: Scores du consensus pondéré par un modèle de langue bigramme.

Nous observons une très nette dégradation des performances. Cela peut s'expliquer par le fait que le modèle de langue est entraîné sur tout le corpus TRAIN et il favorise donc les transitions les plus probables en général et pas du tout en fonction de la phrase à traduire. Au contraire, la pondération précédente en fonction du rang de la phrase à laquelle appartient la transition ne tient compte que d'informations sur les phrases déjà sélectionnées par la mémoire, donc des phrases proches de celle à traduire.

Nous avons aussi essayé de faire une combinaison linéaire des deux pondérations en faisant varier la contribution de chacune mais les performaces de la pondération par le rang sont tellement meilleures que celles de la pondération par le modèle de langue que la meilleure configuration observée était simplement celle avec un coéfficient 1 pour la pondération par le rang et 0 pour la pondération par le modèle de langue.

Avant d'abandonner définitivement la pondération par le modèle de langue, nous avons voulu voir les résultats obtenus avec un modèle trigramme. Ces résultats se trouvent à la table 3.4.

cibles	WER	SER	NIST	BLEU
avant consensus	**22,69%**	**94,82%**	**9,7853**	**66,56%**
consensus bigramme	37,93%	99,68%	7,2500	45,57%
consensus trigramme	36,23%	99,55%	7,4876	47,72%

Table 3.5: Scores du consensus pondéré par un modèle de langue trigramme.

Les scores obtenus avec le modèle trigramme sont légèrement meilleurs que ceux obtenus avec le bigramme mais ils sont encore très insuffisants donc nous avons renoncé à cette approche et nous resterons avec la technique de pondération par le rang.

3.4 Conclusions sur le consensus

Nous avons vu que la technique de consensus proposée par Bangalore *et al.* permet d'améliorer légèrement le wer : nous avons un gain d'au mieux 0,5%. Mais nous avons surtout découvert que les améliorations en termes de ser, nist et bleu sont nettement plus importantes. Toutefois cette technique ne peut être efficace que si les traductions sorties de la mémoire sont assez proches de celle recherchée. Si la phrase à traduire est trop différente de toutes celles qu'il y a dans la mémoire, aucune technique de consensus ne pourra produire la bonne traduction. Nous allons donc nous intéresser maintenant à une autre approche de la traduction automatique : la traduction statistique. Nous espérons que la capacité de cette technique à construire des traductions entièrement nouvelles lui permettra de concurrencer l'approche par mémoire de traduction et d'obtenir de meilleurs résultats sur les phrases qui ne sont pas dans la mémoire.

Chapitre 4
Traduction statistique

Dans les chapitres précédents, nous avons montré comment nous tirons parti efficacement d'une mémoire de traduction au niveau de la phrase. Mais nous avons aussi vu qu'il y aura toujours des phrases à traduire ne ressemblant à aucune phrase dans la mémoire. Pour ces phrases, la mémoire est impuissante. Il nous faut utiliser une technique différente.

Dans ce chapitre, nous allons donc étudier les performances de deux moteurs de traduction statistiques : l'un utilisant un modèle basé sur les mots et l'autre un modèle basé sur les séquences de mots. Nous montrerons que les performances de ces deux moteurs sont très différentes. Le meilleur des deux modèles sera une alternative viable quand la mémoire nous fait défaut.

4.1 Fondements de la traduction statistique

La traduction statistique (ou traduction probabiliste) est une approche proposée dans les années 90 qui se base sur le modèle du canal bruité [2]. L'idée est de voir le problème de la traduction comme un problème de communication : la phrase source que nous voulons traduire peut être vue comme un message qui serait arrivé erroné et que nous souhaitons corriger. Nous voulons retrouver le sens du message, c'est-à-dire sa traduction dans notre langue cible.

Pour ce genre de problème, la théorie de la communication se base sur la loi de Bayes pour énoncer une équation fondamentale :

$$\hat{m} = \mathrm{argmax}_{m \in \mathcal{M}} \; p_T(r|m) \; \times \; p_V(m) \tag{4.1}$$

où m est le message que nous cherchons à reconstituer, $\mathcal{M}$ est l'ensemble des messages possibles, r est le message reçu. $p_T(r|m)$ est la probabilité de recevoir r sachant que c'est m qui a été envoyé. $p_V(m)$ est la probabilité que m soit un message valide.

Dans notre cas, nous traduisons des phrases de l'anglais vers le français. L'équation précédente devient donc :

(4.2)

$$\hat{f} = \text{argmax}_{f \in \mathcal{F}}\ p_T(e|f)\ \times\ p_L(f)$$

où *f* est la traduction recherchée, $\mathcal{F}$ est l'ensemble des phrases françaises possibles et *e* est la phrase anglaise reçue. Cette équation peut être décomposée en trois composantes :

1. le modèle de langue p_L qui nous donne la probabilité qu'une phrase soit du français correct.
2. le modèle de traduction p_T qui nous donne la probabilité qu'une phrase soit la traduction française d'une phrase anglaise donnée.
3. le décodeur (représenté par la fonction argmax dans l'équation) qui trouve la meilleure traduction, c'est-à-dire la phrase qui maximise le produit des probabilités données par les deux modèles.

Notons que la probabilité donnée par le modèle de traduction est la probabilité d'une phrase anglaise étant donnée un phrase française. Bien que nous traduisions de l'anglais vers le français, la phrase source du modèle de traduction est donc en français et sa phrase cible en anglais.

4.1.1 Le modèle de langue

Le modèle de langue nous permet de calculer la probabilité qu'une phrase est correcte en français. Le plus souvent, pour les modèles de langue, une phrase *f* est considérée comme une suite de mots (m_1,m_2,...,m_k). Nous avons alors :

$$p_L(f) = p(m_1|DF) \times p(m_2|DFm_1) \times \ldots \times p(m_k|DFm_1m_2 \ldots m_{k-1}) \quad (4.3)$$

où *DF* représente le début d'une phrase. Clairement, il est impossible de répertorier toutes les probabilités de toutes les séquences de mots possibles dans la langue française. Nous sommes donc obligés de recourir à une approximation markovienne. Une approximation d'ordre 1 nous donne la relation suivante dite relation du bigramme car chaque probabilité ne dépend que de deux mots :

$$p_L^{(2)}(f) = p(m_1|DF) \times p(m_2|m_1) \times \ldots \times p(m_k|m_{k-1}) \quad (4.4)$$

Plus généralement, nous parlerons de modèle *n*-gramme pour une approximation d'ordre $n - 1$:

$$p_L^{(n)}(f) = \prod_{i=1}^{k} p(m_i | m_{i-n+1}^{i-1}) \tag{4.5}$$

où m_{i-n+1}^{i-1} représente la séquence $m_{i-n+1} \ldots m_{i-1}$.

Plus les *n*-grammes d'un modèle sont grands, plus les probabilités qu'il donne seront précises car l'approximation est d'autant plus petite. Mais en échange, plus *n* est grand, plus il y a de combinaisons de mots pour lesquelles notre modèle doit pouvoir donner une probabilité.

La construction d'un modèle de langue peut se faire par simple observation d'un corpus d'entraînement. Un logiciel parcourt le corpus et relève le nombre d'occurrences de tous les *n*-grammes. Il peut ainsi déterminer la probabilité d'un *n*-gramme *N* donné en calculant la fréquence relative de *N*, c'est-à-dire le rapport entre le nombre d'occurrences de *N* et le nombre d'occurrences de tous les *n*-grammes qui commencent par les mêmes $n - 1$ mots que *N*. La formule correspondante est la suivante :

$$p(m_i | m_{i-n+1}^{i-1}) = \frac{|m_{i-n+1}^{i}|}{\sum_m |m_{i-n+1}^{i-1} m|} \tag{4.6}$$

Nous avons déjà produit un modèle de langue trigramme à partir de la partie française de TRAIN afin de réaliser les expériences du chapitre précédent. Nous allons donc réutiliser le même ici. Il s'agit d'un modèle trigramme avec lissage de type JELINEK-MERCER [10] :

$$\begin{aligned} p_{JM}(m_i | m_{i-2} m_{i-1}) &= \lambda_0(m_{i-2} m_{i-1}) \times p(m_i) \\ &+ \lambda_1(m_{i-2} m_{i-1}) \times p(m_i | m_{i-2}) \\ &+ \lambda_2(m_{i-2} m_{i-1}) \times p(m_i | m_{i-2} m_{i-1}) \end{aligned} \tag{4.7}$$

où les coefficients $\lambda_0(m_{i-2}m_{i-1})$, $\lambda_1(m_{i-2}m_{i-1})$ et $\lambda_2(m_{i-2}m_{i-1})$ sont optimisés avec une sous-partie de TRAIN réservée à cet usage. Nous avons aussi :

$$\lambda_0(m_{i-2} m_{i-1}) + \lambda_1(m_{i-2} m_{i-1}) + \lambda_2(m_{i-2} m_{i-1}) = 1 \tag{4.8}$$

pour conserver la norme de la distribution des probabilités.

Ce lissage utilise à la fois des probabilités trigrammes, bigrammes et unigrammes pour éviter de donner des probabilités nulles aux trigrammes qui n'ont jamais été vus dans le corpus d'entraînement. En effet, en raison du chaînage qui est fait avec les probabilités du modèle de langue, une probabilité nulle pour un trigramme entraînerait automatiquement une probabilité nulle pour toute la phrase !

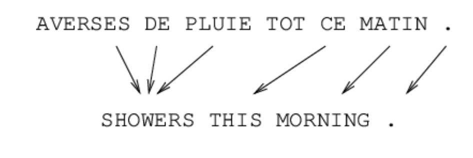
La qualité d'un modèle de langue s'exprime souvent en terme de perplexité. Cette valeur représente le nombre moyen de mots que le modèle va considérer comme pouvant compléter un n-gramme dont les $n - 1$ premiers mots sont fixés. Pour des corpus de language courants (comme par exemple le corpus HANSARD des débats parlementaires canadien), la perplexité des modèles varie typiquement entre 70 et 400. Pour notre corpus météorologique, la perplexité est très bonne car elle est seulement de 4,6. Pour revenir à l'intuition de ce que représente la complexité, ce résultat signifie que si nous donnons à notre modèle les deux premiers mots d'un trigramme et que nous lui demandons, d'après ce qu'il a observé à l'entraînement, de deviner le troisième mot du trigramme, le modèle va généralement hésiter parmi 4 ou 5 mots seulement. Autant dire que notre corpus est assez régulier.

4.1.2 Le modèle de traduction basé sur les mots

Le premier modèle de traduction que nous avons considéré est basé sur les mots (*Word-Based Model* ou WBM). Il s'agit d'un modèle de type IBM 2 décrit par BROWN *et al.* [3]. Nous avons choisi ce type de modèle, le second d'une série de cinq proposés par les auteurs, pour le compromis qu'il offre entre sa simplicité et son efficacité.

Soit E l'ensemble des phrases anglaises et F l'ensemble des phrases françaises. Soit $e = e_1 \ldots e_l$ une phrase de E que nous voulons traduire et $f = f_1 \ldots f_m$ une phrase de F que nous considérons comme traduction potentielle de e. Nous avons dit précédement que pour traduire de l'anglais vers le français, le modèle de traduction doit nous donner la probabilité $p_T(e|f)$. Les modèles de traduction IBM sont tous basés sur la notion d'alignement de mots. Un alignement entre la source f et la cible e est représenté par un vecteur de

positions $a = (a_1,\ldots,a_l)$ avec $\forall i \in [1,l], a_i \in [0,m]$. Chaque a_i est tel que f_i est en relation de traduction avec e_{a_i}. $a_i = 0$ si f_i n'est en relation de traduction avec aucun mot de e. Un tel alignement est représenté à la figure 4.1. Les flèches y indiquent les alignements d'un mot de la source vers un mot de la cible.



a = (1, 1, 1, 0, 2, 3, 4)

Figure 4.1: Alignement entre une phrase française et une phrase anglaise.

Avec cette représentation des alignements, chaque mot de la source est aligné avec au plus un mot de la cible. Par contre, plusieurs mots de la source peuvent être alignés avec le même mot de la cible. Par exemple, nous voyons dans la figure 4.1 que les 3 mots `AVERSES DE PLUIE` sont aligné à `SHOWERS`. Remarquons donc qu'il ne serait pas possible d'aligner correctement les phrases dans l'autre direction (anglais vers français) car `SHOWERS` ne pourrait être aligné qu'à un seul mot. Il s'agit d'une limitation connue des modèles de type IBM.

Un modèle IBM 2 calcule la probabilité que f est la traduction de e par marginalisation de la probabilité jointe $p(e,a|f)$ selon l'équation suivante :

$$p_T(e|f) = \sum_{a \in \mathcal{A}(f,e)} p(e,a|f) \tag{4.9}$$

où $\mathscr{A}(f,e)$ est l'ensemble des alignements valides entre les mots de f vers ceux de e.

Cette probabilité jointe est approchée par deux distributions : la distribution de transfert t et la distribution d'alignement a. Nous avons donc :

$$p(e,a|f) \approx \prod_{i=1}^{l} \sum_{j=0}^{m} (a(j|i,l,m) \times t(e_i|f_j)) \tag{4.10}$$

Les paramètres de ces deux distributions sont appris à partir d'un bitexte d'entraînement à l'aide d'un algorithme itératif décrit dans [3]. Nous avons généré notre modèle de traduction WBM de type IBM 2 grâce à des outils dévelopés au laboratoire RALI[1]. Nous montrons en figure 4.2 un extrait des paramètres appris à partir de notre bitexte TRAIN. Dans cet extrait, on apprend par exemple que APPROCHER est associé à 130 mots dans le bitexte. Les 3 plus fréquents sont listés par ordre de fréquence décroissante.

[1] Giza++, un autre outil pour la génération de modèles de traduction, est disponible sur le site Internet du ISI : http://www.isi.edu/~och/GIZA++.html

```
APPROCHER (3/130):
(APPROACH,0.42) (IS,0.12) (TO,0.12)
AVANCER (3/155):
(MOVE,0.2) (ACROSS,0.13) (FORECAST,0.12)
BANNOCKBURN (2/60):
(BANNOCKBURN,0.51) (KALADAR,0.42)
CAROLINE (3/132):
(CAROLINA,0.49) (OFF,0.3) (COAST,0.15)
CERTAINES (1/180):(SOME,0.65)
ENTRAINERA (2/149):(WILL,0.21) (BRING,0.15)
```

FIGURE 4.2: Extrait du modèle de traduction français-anglais basé sur les mots.

Le lexique probabiliste ainsi obtenu est globalement de bonne qualité même si notre extrait montre les limites de l'hypothèse faite par les modèles IBM sur l'alignement de chaque mot de la source avec un seul mot de la cible. En effet, nous voyons à la dernière ligne de l'extrait que 21% des fois, le modèle associe ENTRAINERA à WILL et 15% des fois, il l'associe à BRING. Cela vient du fait qu'il ne peut pas associer ENTRAINERA aux deux mots à la fois. Ce problème va nous inciter à essayer également un second type de modèle de traduction plus souple sur les alignements.

Avant de passer au second modèle, précisons que pour utiliser ce modèle (c'est-à-dire trouver la phrase cible la plus probable pour une phrase source donnée), nous utilisons une extension du décodeur décrit par NIESSEN *et al.* [23] et réalisé par Philippe LANGLAIS.

4.1.3 Le modèle de traduction basé sur les séquences de mots

Le second type de modèles de traduction que nous avons essayé est basé sur les séquences de mots (*Phrase-Based Model* ou PBM). Dans un modèle PBM, les phrases sont partitionnées en séquence de mots qui sont les paramètres du modèle. Le principal avantage de ces modèles est qu'ils sont capables de capturer des traductions d'expressions entières récurrentes dans le bitexte. La figure 4.3 montre un extrait de notre modèle. Nous y trouvons des traductions d'expressions comme ACROSS CAPE BRETON ↔ AU CAP-BRETON.

```
A FEW GUSTS NEAR ||| QUELQUES RAFALES A PRES ||| 1
A SEA FORECAST ||| PREVISIONS MARITIMES ||| 1
A STATIONARY ||| UN DEPRESSION STATIONNAIRE ||| 0.
5
A LIGHT TROUGH ||| UN FAIBLE CREUX ||| 0.103825
A WEAK TROUGH ||| UN FAIBLE CREUX ||| 0.224044
A WEAK RIDGE ||| UN FAIBLE CRETE ||| 0.204545
ACCUMULATION 5 CM OVER ||| TOTALE DE 5 CM SUR |||
1
ACCUMULATION 5 CM SOUTH ||| TOTALE DE 5 CM SUD |||
 1
ACCUMULATION 5 TO 6 CM ||| TOTALE DE 5 A 6 CM |||
1
ACROSS CAPE BRETON ||| AU CAP-BRETON ||| 0.333333
```

Figure 4.3: Extrait du modèle de traduction français-anglais basé sur les séquences de mots.

Un autre avantage de ces modèles est que le réordonnancement local des mots d'une langue à l'autre est pris en compte de manière passive par un tel modèle. Par exemple, à la seconde ligne de l'extrait, SEA précède FORECAST dans la séquence de mots anglais alors que c'est PREVISIONS qui précède MARITIMES dans la séquence française correspondante. Ce réordonnancement se fait implicitement ici alors qu'il doit être géré par l'alignement des mots dans le modèle IBM 2.

Par contre, l'inconvénient majeur des modèles PBM est la taille de leurs fichiers. En effet, en raison des nombreuses partitions en séquences de mots possibles pour chaque phrase, il n'est pas rare qu'un modèle PBM occupe plus d'un giga-octet de mémoire.

Pour construire notre modèle PBM, nous avons utilisé les outils spécialisés de Philippe Langlais [16]. Ces outils exploitent les alignements obtenus par un modèle de traduction WBM (notre modèle IBM 2 dans ce cas) pour limiter les paires de séquences retenues par le modèle PBM à celles dont certains mots au moins ont été alignés ensembles par le modèle WBM.

Comme décodeur pour ce modèle, nous avons choisi d'utiliser le logiciel pharaoh version 1.2.3 de Philipp Koehn [13] disponible gratuitement sur le site du ISI[2].

[2] http://www.isi.edu/licensed-sw/pharaoh/

4.1.4 Performances des différents moteurs de traduction statistique

Nous présentons dans la table 4.1 les scores des deux engins statistiques sur les phrases hors-mémoire du corpus blanc. Pour comparaison, nous rappelons les scores de la mémoire avant et après consensus sur les mêmes phrases.

candidat	WER	SER	NIST	BLEU
mémoire avant consensus	22,69%	94,82%	9,7853	66,56%
mémoire après consensus	22,97%	85,53%	9,9314	66,86%
modèle de traduction WBM	22,88%	**64,14%**	9,0694	62,03%
modèle de traduction PBM	**15,87%**	66,83%	**11,3310**	**75,77%**

Table 4.1: Scores des moteurs statistiques pour les phrases hors-mémoire.

Les scores obtenus par l'engin PBM sont de loin les meilleurs (sauf pour le ser où le WBM est un peu meilleur). La figure 4.4 montre une traduction produite par l'engin PBM pour une phrase dont la distance avec la mémoire est de 5. L'amélioration est frappante par rapport à la traduction proposée par la mémoire pour la même phrase source (cf. figure 2.8).

Source :
BLIZZARD CONDITIONS WILL BEGIN LATER THAN PREVIOUSLY FORECAST .
Candidate proposée par la mémoire :
ACCUMULATION DE NEIGE SUPERIEURE A CE QUI ETAIT PREVU.
Candidate proposée par l'engin PBM :
BLIZZARD DEBUTERA PLUS TARD QUE PREVU .
Référence :
LE BLIZZARD DEBUTERA PLUS TARD QUE PREVU .

Figure 4.4: Traduction issue de l'engin statistique.

Cela nous confirme donc bien que pour les phrases qui ne ressemblent à aucune phrase dans la mémoire, il vaut mieux produire de nouvelles traductions que d'essayer de construire une traduction par consensus de la sortie de la mémoire. naturellement nous demander s'il ne serait pas possible d'améliorer les traductions produites par l'engin PBM. Pour cela, nous pouvons toujours essayer une approche qui nous est maintenant familière : le consensus.

4.2 Consensus sur la sortie d'un moteur de traduction

4.2.1 Construction du consensus

Puisque le consensus nous a permis d'améliorer les performances de la sortie de la mémoire de traduction, il semble naturel de se demander si le consensus ne pourrait pas avoir le même effet sur la sortie de l'engin PBM.

Dans sa configuration par défaut, PHARAOH ne produit qu'une seule traduction pour chaque phrase source. La construction de cette traduction passe par celle d'un automate similaire à ceux que nous utilisons pour le consensus au chapitre 3, à la différence près que la pondération des arcs est se base sur les probabilités données par le modèle de traduction PBM. La seule traduction retournée par PHARAOH est celle qui correspond au chemin de plus grand poids dans l'automate. Toutefois il est possible de demander au programme de sauvegarder dans un fichier chacun des automates qu'il a utilisé. A partir de chacun de ces fichiers, le logiciel CARMEL (cf. section 3.2.3)

est capable de nous donner les *n* meilleurs chemins dans l'automate. Comme pour la mémoire, nous avons choisi de garder les 10 meilleures traductions différentes pour chaque phrase source.

4.2.2 Performances du consensus

Maintenant que nous avons une sortie de l'engin PBM avec plusieurs traductions candidates, nous pouvons faire un consensus entre ces traductions exactement de la même manière que pour la sortie de la mémoire (cf. section 3.2). Nous avons indiqué à la table 4.2 les scores obtenus avec les différentes pondérations par le rang.

contribution	WER	SER	NIST	BLEU
aucune	**15,87%**	**66,83%**	**11,3310**	**75,77%**
c_0	16,75%	72,94%	11,1955	74,31%
c_1	16,05%	68,93%	11,3010	75,45%
c_2	15,99%	68,41%	11,3130	75,57%
c_3	15,93%	67,82%	11,3207	75,64%
c_4	15,88%	67,29%	11,3271	75,72%
c_5	15,87%	66,91%	11,3297	75,76%

TABLE 4.2: Scores du consensus avec les différentes pondérations par le rang.

Même pour c_5, notre meilleure pondération, les scores après consensus sont légèrement moins bons qu'avant le processus. Cela provient probablement du fait que le décodeur a déjà fait le travail de sélection du meilleur assemblage de groupes de mots pour former la meilleure phrase. En alignant les phrases, reconstruisant un automate et resélectionnant les meilleurs chemins, nous refaisons en quelque sorte le travail de PHARAOH et cela ne nous donne pas de meilleurs résultats. Pour obtenir des phrases vraiment différentes (et peut-être meilleures), il faut introduire dans le consensus des phrases qui proviennent d'une autre source que notre engin PBM, par exemple les traductions qui étaient produites par la mémoire.

4.2.3 Traduction statistique et mémoire

Nous voulons faire un consensus entre plusieurs traductions produites par la mémoire et plusieurs traductions produites par l'engin. Il est difficile de classer toutes ces traductions car leurs mesures de qualité ne sont pas homogènes. La seule chose que nous pouvons donc faire, c'est de mettre toutes les traductions candidates à la suite les unes des autres sans se préoccuper de l'ordre et de faire un consensus en utilisant une pondération constante (c_0) car le rang ne nous donne plus aucune indication de qualité.

La table 4.3 montre les scores obtenus par notre nouveau consensus. Encore une fois, les scores sont moins bons que si nous prenons simplement la sortie de l'engin PBM sans faire de consensus. Cela s'explique probablement par la faible qualité de la sortie de la mémoire sur les phrases hors-mémoire. Il serait probablement plus intéressant de combiner la sortie de plusieurs engins statistiques différents mais de qualité comparable, comme dans l'expérience de Bangalore *et al.* (cf. 3.2). Malheureusement les performances de l'engin WBM sont plutôt comparables à celles de la mémoire. Nous ne disposons pas d'un second engin de traduction aussi performant que l'engin PBM.

candidat	WER	SER	NIST	BLEU
mémoire seule	22,69%	94,82%	9,7853	66,56%
engin PBM seul	**15,87%**	**66,83%**	**11,3310**	**75,77%**
consensus	17,93%	70,84%	11,0216	74,12%

Table 4.3: Scores du consensus entre la mémoire et l'engin statistique.

4.3 Conclusion de la traduction statistique

Dans ce chapitre, nous avons essayé plusieurs techniques de traduction statistique qui ont donné des résultats variés. Nous avons réussi à obtenir des très bons résultats avec le moteur de traduction PBM sur les phrases hors-mémoire alors que le modèle WBM a obtenu des scores plutôt faibles.

Nous avons aussi tenté d’appliquer notre technique de consensus aux sorties de ces deux engins mais le gain de performance à été décevant. Le consensus entre un engin statistique et la mémoire n’a pas été plus concluant. Le moteur de traduction PBM sera donc notre solution de repli quand les différentes variantes de la mémoire ne suffiront pas à traduire une phrase.

Nous avons maintenant tous les éléments nécessaires pour produire un système de traduction performant sur l’ensemble des phrases à traduire. Nous allons passer à l’évaluation finale de notre système.

Chapitre 5
Système final

Dans ce chapitre, nous reprenons chacune des apoproches que nous avons abordées tout au long de notre étude et nous calculons leurs performances sur le corpus TEST qui n'a jamais été utilisé jusqu'à maintenant.

A la vue de ces résultats, nous présentons les choix que nous faisons pour l'architecture de notre système de traduction de bulletins météorologiques. Nous verrons que certaines options de configuration peuvent être laissées à l'utilisateur selon les performances que celui-ci veut maximiser.

5.1 Bilan des différentes parties du système

5.1.1 La mémoire de traduction

Nous avons vu que la mémoire de traduction est une technique simple qui permet d'obtenir des résultats très satisfaisants sur les phrases à traduire que nous avons déjà rencontrées dans le corpus d'entraînement. Pour traduire une phrase, notre système final va donc commencer par rechercher la traduction de la phrase dans la mémoire.

La table 5.1 donne les scores de la mémoire sur toutes les phrases du corpus d'évaluation TEST.

WER	SER	NIST	BLEU
8,42%	23,43%	10,9571	87,68%

TABLE 5.1: Scores de la mémoire sur tout le corpus TEST.

Rappelons seulement que près de 7% du SER est inévitable en raison des inconsistances dans la référence elle-même. Par ailleurs, la valeur minimale de NIST pour TEST est 12 4919.

5.1.2 Le consensus sur la sortie de la mémoire

Si la phrase à traduire n'est pas dans la mémoire, nous avons montré qu'un consensus entre les traductions les plus proches trouvées dans la mémoire donne de bons résultats si la mémoire contient au moins des phrases proches (en terme de distance d'édition) de celle à traduire.

Donc, lorsque la phrase à traduire ne se trouve pas dans la mémoire mais que nous avons assez de phrases suffisamment proches, notre système va se rabattre sur le consensus des différentes traductions proposées par la mémoire.

La table 5.2 donne les scores du consensus sur toutes les phrases du corpus d'évaluation TEST.

WER	SER	NIST	BLEU
12,03%	37,94%	10,5954	82,48%

TABLE 5.2: Scores du consensus sur tout le corpus TEST.

Rappelons que si les scores du consensus sont moins bons que ceux de la mémoire simple, c'est parce que ces scores sont calculés en faisant le consensus sur toutes les traductions produites par la mémoire alors que lorsque la phrase à traduire se trouve telle quelle dans la mémoire, il est préférable de ne pas toucher à la sortie de la mémoire. Notre système n'aura recours au consensus que lorsque les phrases ne se trouvent pas dans la mémoire et nous avons montré que les performances du consensus sur de telles phrases sont meilleures que celles de la mémoire simple.

5.1.3 Le moteur de traduction statistique basé sur les phrases

Enfin, si la phrase à traduire est trop différente, en terme de distance d'édition, de tout ce qu'il y a dans la mémoire, notre système n'aura plus qu'à faire appel à la traduction statistique.

La table 5.3 donne les scores du moteur statistique sur toutes les phrases du corpus d'évaluation TEST.

WER	SER	NIST	BLEU
9,76%	32,01%	10,8725	84,03%

TABLE 5.3: Scores du moteur statistique sur tout le corpus TEST.

5.2 Résultats du système hybride

Notre système dispose donc de trois techniques de traductions ayant chacune leurs points forts et leurs points faibles. La mémoire est efficace pour les phrases déjà vues, le consensus pour les phrases proches de celles déjà vues et le moteur statistique pour les phrases restantes. Notre système a donc deux paramètres : la limite de distance d'édition au-dessous de laquelle le système doit utiliser la mémoire simple et la limite de distance d'édition au-dessus de laquelle notre système doit utiliser le moteur statistique (entre les deux limites se trouve la plage de distances pour laquelle le système utilisera le consensus).

Nous avons essayé toutes les configurations pour ces paramètres variant entre des distance d'édition de 0 et 5. Chaque configuration est designée par une série de 6 lettres parmi M, C ou S selon que c'est respectivement la mémoire, le consensus ou le moteur statistique qui a traduit les

phrases à distance d'édition 0, 1, 2, 3, 4 ou enfin 5 et plus. Les résultats des traductions obtenues sur tout le corpus TEST sont indiqués à la table 5.4. Les meilleurs scores sont indiqués en gras et nous avons également indiqué en italique la ligne ayant les meilleures performances globales.

Configuration						Métrique			
0	1	2	3	4	5+	WER	SER	NIST	BLEU
M	M	M	M	M	M	8,42%	23,43%	10,9571	87,68%
C	C	C	C	C	C	12,03%	37,94%	10,5954	82,48%
S	S	S	S	S	S	9,76%	32,01%	10,8725	84,03%
M	S	S	S	S	S	7,19%	**20,92%**	**11,2859**	89,28%
M	C	S	S	S	S	*7,10%*	*21,54%*	*11,2704*	*89,33%*
M	C	C	S	S	S	7,20%	21,97%	11,2396	89,22%
M	C	C	C	S	S	7,36%	22,24%	11,2045	89,02%
M	C	C	C	C	S	7,48%	22,38%	11,1736	88,88%
M	M	S	S	S	S	**7,06%**	21,98%	11,2614	**89,34%**
M	M	C	S	S	S	7,16%	22,42%	11,2307	89,23%
M	M	C	C	S	S	7,32%	22,69%	11,1956	89,03%
M	M	C	C	C	S	7,44%	22,83%	11,1648	88,89%
M	M	M	S	S	S	7,18%	22,70%	11,2082	89,14%
M	M	M	C	S	S	7,35%	22,97%	11,1733	88,93%
M	M	M	C	C	S	7,47%	23,12%	11,1424	88,79%
M	M	M	M	S	S	7,35%	23,02%	11,1599	88,91%
M	M	M	M	C	S	7,47%	23,16%	11,1291	88,77%
M	M	M	M	M	S	7,47%	23,18%	11,1208	88,77%

TABLE 5.4: Scores des différentes configurations du système hybride.

Les performances de notre système hybride sont clairement meilleures que les performances de ses différentes composantes prises individuellement. Nous sommes

parvenu a tirer profit des différentes qualités de chacune de ces composantes.

Notre système peut être configuré différemment selon le score que nous souhaitons optimiser. Si nous voulons les meilleures performances de WER et BLEU (MMSSSS), il suffit de régler le système pour qu'il garde la sortie de la mémoire pour les phrases à distance 0 ou 1 et qu'il fasse appel au moteur statistique pour les phrases phrases à distance 2 et plus (on ne fait pas de consensus).

Pour les meilleures performances de SER et NIST (MSSSSS), il faut de régler le système pour qu'il garde la sortie de la mémoire pour les phrases à distance 0 seulement et qu'il fasse appel au moteur statistique pour les phrases à distance 1 et plus (on ne fait pas non plus de consensus).

Mais si nous voulons les meilleures performances globales, le meilleur réglage (MCSSSS) consiste à utiliser la mémoire simple pour traduire les phrases à distance 0, faire un consensus entre les phrases issues de la mémoire à distance 1 et sinon d'utiliser le moteur statistique pour produire une nouvelle traduction. Cela sera la configuration proposée par défaut pour notre système de traduction des bulletins météorologiques.

5.3 Evaluation humaine

Lors d'une étude réalisée au CMC en 1985 par Elliott MACKLOVITCH [20], 1 257 phrases anglaises ont été traduites par le système METEO-2 puis soumises à des traducteurs humainrs pour révision. Les résultats de l'étude montrent que 89% des traductions n'ont pas été révisées par les traducteurs.

Nous avons voulu reconduire une évaluation comparable. Nous savons que, dans le réglage MCSSSS, 78,46% des traductions produites par notre système sont identiques aux traductions de référence dans notre bitexte. Ces traductions de référence ayant été validées par des traducteurs humains du CMC, nous supposerons qu'elles sont toutes correctes. Parmi les 21,54% de traductions restantes, nous avons sélectionné aléatoirement 1 000 traductions de phrases différentes et nous avons demandé à des évaluateurs humains de décider si ces traductions sont correctes ou non.

Le résultat de notre évaluation est que 77% des traductions soumises ont été considérées correctes. Si nous ajoutons ces 77% des 21,54% de phrases différentes de la référence aux 78,46% de phrases identiques à la référence, nous obtenons finalement un total de près de 95% de traductions correctes. Il faudrait probablement conduire une plus grande campagne d'évaluation pour pouvoir confirmer ces résultats mais nous pouvons affirmer que notre système a un taux d'erreur au niveau des phrases au moins aussi bas que celui du système original.

Conclusion

Notre étude des textes des bulletins météorologiques a montré qu'il s'agit de textes simples. C'est ce qui explique qu'une approche comme la mémoire de traduction phrastique donne d'aussi bons résultats. Nous avons également montré qu'une approche plus récente comme la traduction statistique basée sur un modèle de traduction des séquences de mots permet d'obtenir des résultats comparables voire meilleurs quand les phrases à traduire ne ressemblent pas à celle rencontrées dans le corpus d'entraînement. Enfin nous avons associé ces différentes techniques au sein d'un même système de traduction qui profite des forces et des faiblesses de chacune.

Nous avons ainsi fait un rapide tour d'horizon des techniques de traduction automatique. Il y a encore d'autres approches que nous n'avons pas du tout abordées comme les modèles de traduction basés sur la syntaxe [25] mais nous avons montré que, avec les approches étudiées, nous pouvons déjà obtenir un système assez performant sur presque toutes les phrases. Nos inplantations de ces approches pourraient aussi probablement être optimisées.

En introduction de notre étude, nous avions entre autres énoncé deux objectifs : améliorer le temps de développement par rapport au système METEO d'origine et aussi améliorer les performances. Le développement de notre système, en incluant la préparation du bitexte, nous a pris moins d'un an. De son côté, le premier système METEO a été développé en deux ans par une grosse équipe de développement. Cette différence s'explique essentiellement par la collection d'outils de traîtement de la langue à notre disposition aujourd'hui (PHARAOH, CARMEL, les outils du

laboratoire RALI...). La vaste quantité de données qui nous a été fournie par le CMC nous a également permi de construire un gros bitext de bonne qualité. Sans cela, aucune de nos expériences n'aurait été possible. Par ailleurs, nous avons montré au chapitre précédent que notre système semble obtenir des résultats comparables, sinon meilleurs, que ceux du système METEO original lorsque que les deux systèmes sont évalués par des humains.

Nous considérons donc que nos deux objectifs initiaux ont été atteints. Nous avons un prototype fonctionnel de système de traduction automatique des bulletins météorologiques. Quels sont les applications possibles de notre système ? Lorsque nous avons présenté notre travail au CMC, une application intéressante nous a été suggérée : la traduction des alertes météorologiques. Les alertes météorologiques sont des bulletins émis spécialement lorsque certains phénomènes météorologiques rares se produisent comme des blizzard ou bien des tempêtes. Ces alertes nécessitent d'être traduites rapidement en raison de l'urgence des informations qu'elles contiennent. De plus, leurs textes sont probablement un peu plus variés que ceux des bulletins traditionnels et le vocabulaire utilisé comporte des termes rares. La partie statistique de notre système pourrait peut-être gérer mieux ces conditions de traduction que le système METEO avec ses grammaires et ses lexiques statiques.

Comme nous l'avons dit plus haut, d'autres approches de traduction automatique sont possibles pour traduire des bulletins météorologiques dans plusieurs langues. Mais la solution la plus simple et efficace n'est probablement pas la traduction automatique mais plutôt la génération automatique de textes multilingues directement à l'origine. Un logiciel s'utilisant comme un formulaire avec des champs à remplir (température, direction du vent, quantité de

précipitation...) pourrait facilement produire automatiquement des bulletins dans plusieurs langues et dans plusieurs formats (texte, HTML, graphiques...). KITTREDGE *et al.* kittredge1998ams présentent un tel logiciel : MeteoCogent[1]. Cette approche nous semble plus naturelle pour les bulletins réguliers qui sont souvent très similaires d'un jour à l'autre et d'une source à une autre. Malheureusement, l'exploitation de la génération automatique de textes souffre de la réticence des météorologues à voir leur liberté de rédaction réduite à des champs numériques et des listes de choix parmi des termes prédéfinis.

1 `http://www.cogentex.com/research/meteocogent/index.shtml`

Index

Bibliographie

[1] BANGALORE, S., BORDEL, G., AND RICCARDI, G. Computing consensus translation from multiple machine translation systems. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop (ASRU-2001)* (ITC Madonna di Campiglio, Trento, Italy, December 2001), IEEE Signal Processing Society, pp. 351–354.

[2] BROWN, P., COCKE, J., PIETRA, S. D., PIETRA, V. D., JELINEK, F., LAFFERTY, J., MERCER, R., AND ROOSSIN, P. A statistical approach to machine translation. *Computational Linguistics 16*, 2 (1990), 79–85.

[3] BROWN, P., PIETRA, S. D., PIETRA, V. D., AND MERCER, R. The mathematics of machine translation : Parameter estimation. *Computational Linguistics 19*, 2 (1993), 263–312.

[4] DODDINGTON, G. Automatic evaluation of machine translation quality using n-gram cooccurrence statistics. In *Proceedings of the Human Language Technology Conference (HLT-2002)* (San Diego, USA, March 2002), pp. 128–132.

[5] FENG, D.-F., AND DOOLITTLE, R. Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *Journal of Molecular Evolution 25* (1987), 351–360.

[6] FOSTER, G., GANDRABUR, S., LANGLAIS, P., PLAMONDON, P., RUSSEL, G., AND SIMARD, M. Statistical machine translation : Rapid development with limited resources. In *Proceedings of MT Summit IX* (New Orleans, USA, September 2003), pp. 110–117.

[7] Grefenstette, G., and Tapanainen, P. What is a word, what is a sentence ? Problems of tokenization. In *Proceedings of the 3rd International Conference on Computational Lexicography (COMPLEX-1994)* (Research Institute for Linguistics, Hungarian Academy of Sciences, Budapest, 1994), pp. 79–87.

[8] Grimaila, A., and Chandioux, J. *Made to Measure Solutions*. In *Computers in Translation : A Practical Appraisal*, Newton, J., Ed., Routledge, London, UK, 1992, ch. 3, pp. 33–45. John Newton Ed.

[9] Hutchins, J. The origins of the translator's workstation. *Machine Translation 13*, 4 (1998), 287–307.

[10] Jelinek, F., and Mercer, R. Interpolated estimation of markov source parameters from sparse data. In *Proceedings of the Workshop on Pattern Recognition in Practice* (Amsterdam, Netherlands, 1980), pp. 381–397.

[11] Kittredge, R., and Lavoie, B. MeteoCogent : A knowledge-based tool for generating weather forecast texts. In *Proceedings of the 1st American Meteorological Society Conference on Artificial Intelligence (AMS-1998)* (Phoenix, USA, January 1998), pp. 80–83.

[12] Knight, K., and Al-Onaizan, Y. *A Primer on Finite-State Software for Natural Language Processing*, August 1999. http ://www.isi.edu/licensed-sw/carmel/carmel-tutorial2.pdf.

[13] Koehn, P. Pharaoh : A beam search decoder for phrase-based statistical machine translation models. In *Proceedings of the 6th Conference of the Association for Machine Translation in the Americas (AMTA-2004)* (Georgetown University, Washington DC, USA, September 2004), Association for Machine Translation in the Americas, pp. 115–124.

[14] Koehn, P. *Pharaoh : User Manual and Description for Version 1.2*, August 2004. http ://www.isi.edu/licensed-sw/pharaoh/manual-v1.2.ps.

[15] Langlais, P. Alignement de corpus bilingues : intérêts, algorithmes et évaluations. *Bulletin de Linguistique Appliquée et Générale*, Hors-série (1997), 245–254.

[16] Langlais, P., Carl, M., and Streiter, O. Experimenting with phrase-based statistical translation within the IWSLT 2004 Chinese-to-English Shared Translation Task. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT-2004)* (ATR Spoken Language Translation Research Laboratories, Kyoto, Japan, September 2004), pp. 31–38.

[17] Langlais, P., and Simard, M. Merging example-based and statistical machine translation : An experiment. In *Proceedings of the 5th Conference of the Association for Machine Translation in the Americas (AMTA-2002)* (Tiburon, USA, October 2002), Association for Machine Translation in the Americas, pp. 104–113.

[18] Leplus, T., Langlais, P., and Lapalme, G. A corpus-based approach to weather report translation. In *Proceedings of Computational Linguistics in the North-East (CLiNE-2004)* (Concordia University, Montréal, Canada, August 2004), pp. 9–15.

[19] Leplus, T., Langlais, P., and Lapalme, G. Weather report translation using a translation memory. In *Proceedings of the 6th Conference of the Association for Machine Translation in the Americas (AMTA-2004)* (Georgetown University, Washington DC, USA, September 2004), Association for Machine Translation in the Americas, pp. 154–163.

[20] Macklovitch, E. A linguistic performance evaluation of METEO 2. Tech. rep., Translation Bureau of the Canadian

Meteorological Center, August 1985.

[21] Macklovitch, E. Machine-aided translation. To be published in Encyclopedia of Language and Linguistics, 2nd Ed., Elsevier, 2005.

[22] Macklovitch, E., and Russell, G. What's been forgotten in translation memory. In *Proceedings of the 4th Conference of the Association for Machine Translation in the Americas (AMTA-2000)* (Cuernavaca, Mexico, September 2000), Association for Machine Translation in the Americas, pp. 137–146.

[23] Neissen, S., Vogel, S., Ney, H., and Tillmann, C. A dp based search algorithm for statistical machine translation. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING-1998)* (Université de Montréal, Montréal, Canada, August 1998), pp. 960–967.

[24] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. Bleu : a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics (ACL-2002)* (Philadelphia, USA, July 2002), pp. 311–318.

[25] Schafer, C., and Yarowski, D. Statistical machine translation using coercive two-level syntactic transduction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)* (Sapporo, Japan, July 2003), Association for Computational Linguistics, pp. 9–16.

[26] Simard, M., and Langlais, P. Sub-sentential exploitation of translation memories. In *Proceedings of MT Summit VIII* (Santiago de Compostela, Spain, September 2001), pp. 18–23.

[27] Thompson, J., Higgins, D., and Gibson, T. CLUSTAL W : Improving the sensitivity of progressive multiple

sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research 22*, 22 (1994), 4673–4680.